Modelling Description Language

Nick Holford and Mike K Smith

Drug Disease Model Resources

"Builds and maintains a universally applicable, open source, model based framework, intended as the gold standard for future collaborative drug and disease Modelling & Simulation"

http://www.ddmore.eu/content/project

MDL Structure

Model Coding Language

Task Execution Language

Language

Object Group

Model

Task Object Group

Objects

Model Coding Object Group

Task Execution Object Group

Modelling Description Language

A single, unified language to capture the definition of models and modelling tasks

Disease Model Library

A repository of specific disease area models to make quantitative knowledge accessible and sharable

Modelling Framework

A modelling platform for integrating tools and reasoning across to remove barriers and promote transparency

Education & Training - build excellence and foster adoption

Standards & Ontologies

to describe models, data, variables, reactions, tasks and domains

Systems Interchange Standards

http://www.ddmore.eu/content/project
Model Coding Language – Model Object 1

```plaintext
GROUP_VARIABLES:
  GRPCL = POP_CL*(WT/70)^0.75
  GRPV = POP_V*WT/70
  GRPKA = POP_KA
  GRPLG = POP_TLAG

INDIVIDUAL_VARIABLES:
  CL = GRPCL*exp(eta_PPV_CL)
  V = GRPV*exp(eta_PPV_V)
  KA = GRPKA*exp(eta_PPV_KA)
  ALAG1 = GRPLG*exp(eta_PPV_TLAG)
```

Model Coding Language – Model Object 2

```plaintext
RANDOM_VARIABLE_DEFINITION:
  eta_PPV_CL ~ (type=Normal, mean=0, variance=PPV_CL, level=ID)
  eta_PPV_V ~ (type=Normal, mean=0, variance=PPV_V, level=ID)
  eta_PPV_KA ~ (type=Normal, mean=0, variance=PPV_KA, level=ID)
  eta_PPV_TLAG ~ (type=Normal, mean=0, variance=PPV_TLAG, level=ID)
  eps_RUV_PROP ~ (type=Normal, mean=0, variance=RUV_PROP, level=ID)
  eps_RUV_ADD ~ (type=Normal, mean=0, variance=RUV_ADD, level=ID)
```

Model Coding Language – Model Object 3

```plaintext
LIBRARY:
  PPK(input=first-order, distribution=1, parameterization=vcl-k, param=list(cl=CL, v=V, tlag=ALAG1, ka=KA))

CONC = F.A1/V

OBSERVATION:
  Y = CONC*(1+eps_RUV_PROP)+eps_RUV_ADD

OUTPUT:
  ID
  TIME
  Y
```
Evolutionary step from NM-TRAN

Model Coding Language – Data Object

```plaintext
warfarin_PK_CONC.par = parobj
|
| STRUCTURAL
| POP_CL=list(value=0.1,lo=0.001)
| POP_V=list(value=8,lo=0.001)
| POP_KA=list(value=2,lo=0.001)
| POP_TLAG=list(value=1,lo=0.001)
| |
| VARIABILITY
| matrix
| PPV_CL=0.1, 0.01
| PPV_V=0.1
| diag
| PPV_KA=0.1
| |
| RUV_PROP=list(value=0.01)
| |
| RUV_ADD=list(value=0.05, units="mg/L"
| |
| OMEGA BLOCK(2)
| 0.1 ; PPV_CL
| 0.01 0.1 ; PPV_V
| |
| SIGMA
| 0.01 ; RUV_PROP
| 0.05 ; RUV_ADD mg/L
```

Model Coding Language – Parameter Object

```plaintext
warfarin_PK_CONC.par = parobj
|
| STRUCTURAL
| POP_CL=list(value=0.1,lo=0.001)
| POP_V=list(value=8,lo=0.001)
| POP_KA=list(value=2,lo=0.001)
| POP_TLAG=list(value=1,lo=0.001)
| |
| VARIABILITY
| matrix
| PPV_CL=0.1, 0.01
| PPV_V=0.1
| diag
| PPV_KA=0.1
| |
| RUV_PROP=list(value=0.01)
| |
| RUV_ADD=list(value=0.05, units="mg/L"
| |
| OMEGA BLOCK(2)
| 0.1 ; PPV_CL
| 0.01 0.1 ; PPV_V
| |
| SIGMA
| 0.01 ; RUV_PROP
| 0.05 ; RUV_ADD mg/L
```

Model Coding Language – Task Properties Object

```plaintext
warfarin_PK_CONC.task = taskobj
|
| DATA IGNORE=if(DVID==2)
| |
| myEST=functoin(t,m,p,d)
| |
| ESTIMATE
| target=t
| model=m
| parameter=p
| data=d
| algo=list("COND  INTER")
| max=9990
| sig=3
| cov="y"
| |
| PYTHON
| (0.001,0.1) ; POP_CL L/h/70kg
| (0.001,8)   ; POP_V L/70kg
| (0.001,2) ; POP_KA h-1
| (0.001,1) ; POP_TLAG h
| |
| JAGS
| (0.001,0.1) ; POP_CL L/h/70kg
| (0.001,8)   ; POP_V L/70kg
| (0.001,2) ; POP_KA h-1
| (0.001,1) ; POP_TLAG h
| |
| R
| 0.01 ; RUV_PROP
| 0.05 ; RUV_ADD mg/L
```

Evolutionary step from NM-TRAN

Model Coding Language – Data Object
Task Execution Language

- MCL = Nouns, TEL = Verbs, MCL Task Properties = Adverbs.

GET <<Model + Data + Parameter initial values>> and
DO <<Estimation>>
(LIKE THIS <<Task Properties>>) 

- Tasks define what MCL objects are required:
  - Estimation = Model + Data + Parameters (initial, bounds) + Task Properties (Settings)
  - Simulation / Optimal Design = Model + Data Design + Parameters (point estimates or distributions) + Task Properties (Settings)

Evolutionary step from NM-TRAN

Model Coding Language – Task Properties Object

```
warfarin_PK_CONC_task = taskobj{
  DATA{IGNORE=if(DVID==2)}
  myEST=function(t,m,p,d){
    ESTIMATE{
      target=t
      model=m
      parameter=p
      data=d
      algoist("COND INTER")
      max=9990
      sig=3
      cov="y"
    }
  }
}
```

- Windows Command line using Wings for NONMEM
  - Fit model using NONMEM
    `nmgo warfarin_PK_CONC`
  - Update parameter estimates with final estimates
    `nmctl warfarin_PK_CONC`

Evolutionary step from WFN, PsN, etc.

Task Execution Language – TEL Command Object

```
warfarin_PK_CONC_tel = telobj{
  # Fit model using NONMEM
  warfarin_PK_CONC_fit = myEST(t="NONMEM",
    NONMEM= warfarin_PK_CONC_mdl,
    warfarin_PK_CONC_par,
    warfarin_PK_CONC_dat)
  # Update parameter estimates with final estimates
  warfarin_PK_CONC_parupdate(warfarin_PK_CONC_fit, warfarin_PK_CONC_par)
}
```

- TEL defines basic tasks that can build to more complex workflows.
Use R for general data and statistical tasks

Task Execution Language – R Command Object

Your favourite R script

Translation to other languages

"Rosetta Stone"

MDL (alternative random effects)

MLTRAN

BUGS

Phoenix Modeling Language (PML)

GROUP_VARIABLES{
  GRPV = POP_V*WT/70
  ...
}

RANDOM_VARIABLE_DEFINITION{
  lnV ~ (type=Normal, mean=log(GRPV), variance=PPV_V, level=ID)
  ...
}

INIDIVIDUAL_VARIABLES{
  V = exp(lnV)
  ...
}

EQUATION:
  GRPV = POP_V*(WT/70)
  ...
}

DEFINITION:
  V = {distribution=logNormal, prediction=GRPV, standardDeviation=PPV_V}
  ...
}

LOG.GRPV = log(POP_V) + log(WT/70)

for (i in 1:N){
  lnV[i] ~ dnorm(LOG.GRPV, PPV_V)
  ...
}

V[i] = exp(lnV[i])

MDL and the rest of DDMoRe

Standards & Ontologies

Modelling Description Language

A single, unified language to encapsulate the definition of models and modeling tasks

Modelling Framework

A standard platform for integrating tools and modeling models to restore barriers and promote consistency

Disease Model Library

A collection of specific disease and models to make specific models and knowledge accessible and usable

Education & Training

To build excellence and foster adoption
MDL and the rest of DDMoRe

Model Coding Language
- Modular models combining library functions
- Levels of random effect
- Non-normal distributions
- "odd type data" statements
  - POISSON
  - CATEGORICAL
  - HAZARD

Task Execution Language
- Target software appropriate to task using the same model
- Mix and match using library modelling object groups
- Workflow of tasks through framework

Active engagement with target software developers
- ICON on future developments in NONMEM
- Pharsight considering using MDL to enhance PML
- Metrum on implementation with BUGS.

Valuable discussion and input from DDMoRe participants
- Subject matter experts contributing to MCL language features & TEL task definitions.
Modelling Object Group (MOG)

- MOG objects
  - Model, parameter, data, task properties
  - Required inputs to TEL task object
  - The ”model” is the MOG
  - User may combine objects from Repository MOG with user objects
    - E.g. Repository model+parameter with user data+task properties

- MOG Types
  - Defined & curated, static, (public)
  - User defined, read/write, (private, group, public)

Some Ways to Use a MOG

- Full Model (D,P,M,T)
  - Run the model to verify previous results

- Model (M)
  - Library call for model predictions (mixed effect)
  - User supplied D, P, T

- Simulate(M,P)
  - User supplied D and T

- Estimate(D,P,M)
  - User supplied estimation T using library D

- Data Transform(D)
  - User supplied T to transform library D

D=data, P=parameter, M=model, T=task_properties
MDL PK Library Function 1

A one compartment model with first-order input and first-order elimination. Dose is administered to compartment zero. Central compartment is 1 even if input is changed to zero-order or bolus.

LIBRARY {
  F=PK(input=first-order, distribution=1, elimination=first-order, parameterization=vcl-k,
       param=list(cl=CL, v=V, DCMT=0, # input (depot) compartment is 0
                  tlag=ALAG0, ka=KA
       ))

  CONC=F.A1/V
}

MDL PK Library Function 2

A one compartment model with bolus input and parallel first and mixed-order elimination. The first-order elimination pathway is the formation route for a metabolite. The metabolite disposition is described by two compartments with mixed-order elimination. The metabolite has a delayed effect described by an effect compartment linked to the metabolite compartment.

LIBRARY {
  F=PK(input=bolus, distribution=1, elimination=parallel-first-mixed-order,
        metabolite-formation=first-order, metabolite-distribution=2, metabolite-
        elimination=mixed-order, metabolite-link=effect, parameterization=vcl-t,
        param=list(v1=10, clfo=1, vmax=3, km=1, # parent
                    FCMT=1, # metabolite is formed from 1st compartment of parent
                    CLMT=1, # Assume all first-order parent elimination leads to metabolite formation
                    v1m=10, vmaxm=2, km=m=1, v2m=100, cl2m=4, # metabolite
                    LCMTm=1m, tegm=1, # effect is determined by linking to metabolite compartment 1
       ))
}

cem=F.effectm # effect compartment concentration of metabolite

effect=emax*cem/(c50+cem) # effect of metabolite

22

23