merlin: Mixed effects regression for linear, non-linear and user-defined models

Stata Nordic and Baltic Meeting
Oslo, 12th September 2018

Michael J. Crowther

Biostatistics Research Group,
Department of Health Sciences,
University of Leicester, UK,
michael.crowther@le.ac.uk
@Crowther_MJ

Funding: MRC (MR/P015433/1)
the plan

- the motivation
- the past
- the goal
- the example
- the family
- the surprise (at least it was last week)
- the future
the motivation

- More data \(\rightarrow\) more questions
  - need for appropriate statistical modelling techniques, and implementations
the motivation

- More data $\rightarrow$ more questions
  - need for appropriate statistical modelling techniques, and implementations
- Growth in access to EHR
  - biomarkers $<$ patients $<$ GP practice area $<$ geographical regions...
the motivation

- More data $\rightarrow$ more questions
  - need for appropriate statistical modelling techniques, and implementations
- Growth in access to EHR
  - biomarkers $<$ patients $<$ GP practice area $<$ geographical regions...
- The standard challenges
  - time-dependent effects, non-linear covariate effects
the motivation

- More data → more questions
  - need for appropriate statistical modelling techniques, and implementations
- Growth in access to EHR
  - biomarkers < patients < GP practice area < geographical regions...
- The standard challenges
  - time-dependent effects, non-linear covariate effects
- The neglected challenges
  - Within-patient variability
  - Informative observations times
the motivation

• More data $\rightarrow$ more questions
  • need for appropriate statistical modelling techniques, and implementations

• Growth in access to EHR
  • biomarkers $<$ patients $<$ GP practice area $<$ geographical regions...

• The standard challenges
  • time-dependent effects, non-linear covariate effects

• The neglected challenges
  • Within-patient variability
  • Informative observations times

We need modelling frameworks that can accommodate a lot of different things
Joint longitudinal-survival models

Linking via - current value, gradient, AUC, random effects...
Joint longitudinal-survival models - extensions

- Competing risks
- Different types of outcomes
- Multiple continuous outcomes
- Delayed entry
- Recurrent events and a terminal event
- Prediction
- Many others...
Joint longitudinal-survival models - software

- stjm in Stata
- gsem in Stata
- frailtypack in R
- joineR in R
- JM and JMBayes in R
- Many others...
(My) Methods development - software

- stjm - joint longitudinal-survival models
- stmixed - multilevel survival models
- stgenreg - general parametric survival models
- ...

(My) Methods development - software

- stjm - joint longitudinal-survival models
- stmixed - multilevel survival models
- stgenreg - general parametric survival models
- ...

Each new project brings a new code base to maintain...could I make my life easier?
the past

• last year I introduced megenreg
• megenreg fitted mixed effects generalised regression models
• megenreg was awesome...but
the past

• last year I introduced `megenreg`
• `megenreg` fitted mixed effects generalised regression models
• `megenreg` was awesome...but

I really hated the name
Michael Crowther @Crowther_MJ · Apr 16
In the midst of a rewrite of the #meqenreg engine, plus lots of extensions. Building up to release makes me think a rebrand is needed...

71% merlin
14% forge
7% meregress
8% Keep thinking...

14 votes · Final results
Some people were not so keen...

I think FORGE is better than MERLIN because that could sound a bit like it's coming from a nerd who likes playing fantasy games in mum's basement!

Mar 28
Mixed Effects Regression for Linear, Non-linear and user-defined models

merlin
the goal

- multiple outcomes of varying types
- measurement schedule can vary across outcomes
- any number of levels and random effects
- sharing and linking random effects between outcomes
- sharing functions of the expected value of other outcomes
- a reliable estimation engine
- easily extendable by the user
- ...

a unified framework for data analysis and methods development
the example

- there’s no equations in this talk
- there’s 14 models
- each of them is applied to the same dataset
- most of them can be considered *new* models
- we can fit all of them with a single line of code
• data from 312 patients with PBC collected at the Mayo Clinic 1974-1984 (Murtaugh et al. (1994))
• 158 randomised to receive D-penicillamine and 154 to placebo
• survival outcome is all-cause death, with 140 events observed
  • we’re going to pretend we have competing causes of death - cancer and other causes
• 1945 measurements of serum bilirubin, among other things
### The Data

<table>
<thead>
<tr>
<th>id</th>
<th>time</th>
<th>logb</th>
<th>prothr-n</th>
<th>trt</th>
<th>stime</th>
<th>cancer</th>
<th>other</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2.674149</td>
<td>12.2</td>
<td>D-penicil</td>
<td>1.09517</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.525682</td>
<td>3.058707</td>
<td>11.2</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0.0953102</td>
<td>10.6</td>
<td>D-penicil</td>
<td>14.1523</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.498302</td>
<td>-.2231435</td>
<td>11</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>0.999343</td>
<td>0</td>
<td>11.6</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>2.10273</td>
<td>0.6418539</td>
<td>10.6</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>4.90089</td>
<td>0.9555114</td>
<td>11.3</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>5.88928</td>
<td>1.280934</td>
<td>11.5</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>7.8907</td>
<td>1.280934</td>
<td>.</td>
<td>D-penicil</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>
a model

merlin (logb
  time
, family(gaussian)
)

/// log serum bilirubin
/// covariate
/// options
/// distribution
a model

merlin (logb
time
time#trt,
family(gaussian)
)

/// log serum bilirubin
/// covariate
/// interaction
/// options
/// distribution
///
a model

merlin (logb
  time
  time#trt
  M1[id]@1
  ,
  family(gaussian)
  )
  /// log serum bilirubin
  /// covariate
  /// interaction
  /// random intercept
  /// options
  /// distribution
  ///
a model

merlin (logb

time

time#trt

M1[id]@1

time#M2[id]@1

,)

family(gaussian)

/// log serum bilirubin

/// covariate

/// interaction

/// random intercept

/// random slope

/// options

/// distribution
a model

merlin (logb
  time
  time#trt
  M1[id]@1
  time#M2[id]@1,
  family(gaussian)
)

(pro
  rcs(time, df(3))
  , family(gamma)
)

/// log serum bilirubin
/// covariate
/// interaction
/// random intercept
/// random slope
/// options
/// distribution
/// prothrombin index
/// covariate
/// distribution
a model

merlin (logb


time

time#trt

M1[id]@1

time#M2[id]@1

, family(gaussian)

)

(pro

rcs(time, df(3))

M3[id]@1

, family(gamma)

)

/// log serum bilirubin

/// covariate

/// interaction

/// random intercept

/// random slope

/// options

/// distribution

/// prothrombin index

/// covariate

/// random effect

/// distribution

///
merlin (logb
    time
    time#trt
    M1[id]@1
    time#M2[id]@1
    ,
    family(gaussian)
)
( pro
    rcs(time, df(3))
    M3[id]@1
    , family(gamma)
)
,
    covariance(unstructured)

/// log serum bilirubin
/// covariate
/// interaction
/// random intercept
/// random slope
/// options
/// distribution
/// prothrombin index
/// covariate
/// random effect
/// distribution
/// main options
// vcv
a model

merlin (logb
  time
  time#trt
  M1[id]@1
  time#M2[id]@1
  ,
  family(gaussian)
)
(pro
  rcs(time, df(3))
  M3[id]@1
  , family(gamma)
)
,
covariance(unstructured)
redistribution(t) df(5)
/// log serum bilirubin
/// covariate
/// interaction
/// random intercept
/// random slope
/// options
/// distribution
/// prothrombin index
/// covariate
/// random effect
/// distribution
///
/// main options
/// vcv
/// re dist.
a model

merlin (logb
  time
  time#trt
  M1[id]@1
  time#M2[id]@1
  ,
  family(gaussian)
)

(pro
  rcs(time, df(3))
  M3[id]@1
  , family(gamma)
)

(stime trt
  , family(rp, df(3)
    failure(other))
)

,
covariance(unstructured)
covariance(unstructured)
redistribution(t) df(5)
a model

merlin (logb

  time
  time#trt
  M1[id]@1
  time#M2[id]@1
  ,
  family(gaussian)
)

(pro

  rcs(time, df(3))
  M3[id]@1
  , family(gamma)
)

(stime trt

  dEV[logb] EV[pro]
  , family(rp, df(3)
    failure(other))
)

,

covariance(unstructured)
  redistribution(t) df(5)

/// log serum bilirubin
/// covariate
/// interaction
/// random intercept
/// random slope
/// options
/// distribution
/// prothrombin index
/// covariate
/// random effect
/// distribution
/// response + covariate
/// associations
/// distribution
/// event indicator
/// main options
/// vcv
/// re dist.
**a model**

merlin (logb

  time
  time#trt
  M1[id]@1
  time#M2[id]@1
  
  family(gaussian)

)

(pro

  rcs(time, df(3))
  M3[id]@1
  
  family(gamma)

)

(stime

  trt
  trt#fp(stime, power(0))
  dEV[logb] EV[pro]
  
  family(rp, df(3)
  
  failure(other))

)

, covariation(unstructured)

redistribution(t) df(5)

/// log serum bilirubin

/// covariate

/// interaction

/// random intercept

/// random slope

/// options

/// distribution

/// prothrombin index

/// covariate

/// random effect

/// distribution

/// response + covariate

/// tde

/// associations

/// distribution

/// event indicator

/// main options

/// vcv

/// re dist.
a model

merlin (logb time time#trt M1[id]@1 /// model 1
time#M2[id]@1 , ///
family(gaussian) ///
)
(pro rcs(time, df(3)) M3[id]@1 /// model 2
 , family(gamma) ///
)
(stime trt ///
trt#fp(stime, power(0)) /// model 3 - cause 1
dEV[logb] EV[pro] /// tde
 , family(rp, df(3) /// distribution
 failure(other)) /// event indicator
)
(stime trt /// model 4 - cause 2
trt#rcs(stime, df(3) log) /// tde
 EV[logb] iEV[pro] /// associations
 , family(weibull, /// distribution
 failure(cancer)) /// event indicator
)
,
covariance(unstructured)
predictions

predict cif1, cif marginal outcome(3) at(trt 0)
predict cif1, cif marginal outcome(4) at(trt 0)
a user-defined model

```plaintext
real matrix gauss_logl(gml)
{
    y = merlin_util_depvar(gml)    // dep. var.
    linpred = merlin_util_xzb(gml) // lin. pred.
    sdre = exp(merlin_util_ap(gml,1))    // anc. param.
    return(lnnormalden(y,linpred,sdre)) // logl
}

merlin (logb ... , family(user, llfunction(gauss_logl) nap(1)))
... 
... 
... 
```
a user-defined model

real matrix gauss_logl(gml)
{
    y = merlin_util_depvar(gml)      // dep. var.
    linpred = merlin_util_xzb(gml)   // lin. pred.
    sdre = exp(merlin_util_xzb_mod(gml,2))  // anc. param.
    return(lnnormalden(y,linpred,sdre))  // logl
}

merlin (logb ... , family(user, llfunction(gauss_logl)))
   (age M1[id]@1, family(null))
   ...
   ...
**a user-defined nonlinear model - Yulia’s talk**

```stata
webuse orange, clear
menl circumf = (b1+U1[tree])/(1+exp(-(age-b2)/b3))

mata:
real matrix logl(transmorphic gml)
{
    y = merlin_util_depvar(gml)
    b1 = merlin_util_xzb(gml)
    b2 = merlin_util_xzb_mod(gml,2)
    b3 = merlin_util_xzb_mod(gml,3)
    sdre = exp(merlin_util_ap(gml,1))
    xb = b1 :/ (1 :+ exp(-b2 :/ b3))
    return(lnnormalden(y,xb,sdre))
}
end

merlin (circumf M1[tree]@1, family(user, llf(logl) nap(1)))
    ( age@1 , family(null))
    ( , family(null))
```

Michael J. Crowther  merlin  12th September 2018
stuff I didn’t show

- random effects at arbitrary levels - M4[centre>id]@1
- B-splines - bs(time, df(3) order(4))
- d2EV[], ?XB[]
- linterval(varname) - interval censoring
- ltruncated(varname) - left-truncation
- 9 (so far) other inbuilt families, e.g. beta, ologit
- bhazard(varname) - relative survival
- mf(func_name) - user-defined element function
the family

- merlin’s syntax is not simple
- we can develop more user-friendly shell files to allow a simpler syntax for special cases
- merlin’s minions...
  - excalibur (stmixed) for multilevel survival analysis (SJ under revision)
  - lancelot - meta-analysis
  - arthur - to be revealed next!
  - galahad - maybe next year
  - ...

Michael J. Crowther
Two useful features of merlin are:

- \( EV[\text{depvar/#}] \) element type
  - implemented for their use in joint longitudinal-suvival models
- \( \text{family}(\text{null}) \)
  - implemented for use with user-defined models

Their combination gives merlin some new capabilities.
the surprise

merlin \((y \ x1 \ x2 \ EV[2] \ EV[3], \ family(bernoulli) \ link(logit))\)
\((x1 \ x2, \ family(null) \ link(logit))\)
\((x1 \ x2, \ family(null) \ link(logit))\)

any idea what this is?
the surprise

merlin (y x1 x2 EV[2] EV[3], family(bernoulli) link(logit))
(x1 x2, family(null) link(logit))
(x1 x2, family(null) link(logit))

any idea what this is?

It’s an artificial neural network!
**Title**

**neuralnet** — fit an artificial neural network

**Syntax**

```
neuralnet [varlist] , options
```

where `varlist` defines any inputs to the network.

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>output#(depvar, op_opts)</code></td>
<td>output model specification; see details</td>
</tr>
<tr>
<td><code>hlayers(#)</code></td>
<td>number of hidden layers in the network</td>
</tr>
<tr>
<td><code>hlink(link_list)</code></td>
<td>link functions for each hidden layer to the layer above</td>
</tr>
<tr>
<td><code>hnodes(numlist)</code></td>
<td>number of nodes per hidden layer</td>
</tr>
<tr>
<td><code>penalty(pen_func)</code></td>
<td>penalty function; <code>lasso</code> or <code>ridge</code></td>
</tr>
<tr>
<td><code>lambda(#)</code></td>
<td>penalty parameter value; default 0.1</td>
</tr>
<tr>
<td><code>nostandardise</code></td>
<td>do not standardise input variables to $[0,1]$</td>
</tr>
<tr>
<td><code>loss</code></td>
<td>minimise the loss function instead of maximising the log-likelihood</td>
</tr>
<tr>
<td><code>showmerlin</code></td>
<td>displays the <code>merlin</code> command used in estimating the network</td>
</tr>
<tr>
<td><code>merlin_opts</code></td>
<td>options to pass to <code>merlin</code></td>
</tr>
</tbody>
</table>

**output options**

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>family(fam_spec)</code></td>
<td>distributional family for the output/response, see <code>merlin families</code></td>
</tr>
<tr>
<td><code>link(type)</code></td>
<td>link function for the response model</td>
</tr>
</tbody>
</table>
the surprise

merlin (y x1 x2 EV[2] EV[3], family(bernoulli) link(logit))
(x1 x2, family(null) link(logit))
(x1 x2, family(null) link(logit))

neuralnet x1 x2, output1(y, family(bernoulli) link(logit))
hlayers(1) hlink(logit) hnodes(2)
penalty(ridge) lambda(1e-07)
the surprise

       , family(bernoulli) link(logit))
     (x1_nn x2_nn, family(null) link(atanh))
     (x1_nn x2_nn, family(null) link(atanh))
     (EV[2] EV[3], family(null) link(atanh))
     (EV[2] EV[3], family(null) link(atanh))
     (EV[2] EV[3], family(null) link(atanh))

neuralnet x1 x2, output1(y, family(bernoulli) link(logit))
   hlink(atanh) hlayers(2) hnodes(2 3)
   penalty(lasso) lambda(1e-07)
Artificial neural network
From my website - I’m now a data scientist!

**Interests**

- Survival Analysis
- Multilevel Models
- Joint Modelling
- **Machine Learning**
- Software Development
the future

- merlin can do a lot of things, hopefully in a usable way
- merlin is easily extended
- I continue to discover more and more things it can do
- arthur (neuralnet)
  - It's a rubbish implementation of neural networks
  - Needs analytic gradients to be useful
  - penalisation
  - But - all capabilities of merlin can be used in a neural network, and vice versa
- predict newvar, statistic ci

www.mjcrowther.co.uk/software/merlin
the papers

the reversal

I've just realised that Merlin is the better name...

The syllables start with M & L, which represents maximum likelihood and machine learning!

Ah man you've just added to the t-shirts I can have made 😊