merlin: A framework and implementation(s) for extended mixed effects regression of linear, non-linear and user-defined models

Michael J. Crowther

Associate Professor of Biostatistics
Biostatistics Research Group
Department of Health Sciences
University of Leicester, UK
and
Department of Medical Epidemiology and Biostatistics
Karolinska Institutet, Sweden

University of Innsbruck
9th January 2020

mjcrowther.co.uk michael.crowther@le.ac.uk @Crowther_MJ

Outline

- My background
- Motivation for this work
- Extended multivariate generalised linear and non-linear mixed effects models
- `merlin`
- Methods development using `merlin`
- Future directions

My background

- MMath in Maths and Stats at University of St. Andrews, graduating in May 2009
- Awarded PhD in November 2014, titled "Development and application of methodology for the parametric analysis of complex survival and joint longitudinal-survival data in biomedical research"
- One year post-doc at Karolinska Institutet in Stockholm
- Appointed a Lecturer in March 2016 at Leicester
- Became Associate Professor in August 2018
- Now 20% seconded to Karolinska Institutet from this month

My background

- I mainly work in methodology; developing new statistical models and software
- My applied work is generally in cardiovascular disease and cancer
- My main areas of research are:
  - joint modelling of longitudinal and survival data
  - multi-state survival analysis
  - software development
- Development of methods for complex survival analysis, MRC NIRG Methodology Research Panel, March 2017 - March 2020

Motivation

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

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...

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

Motivation

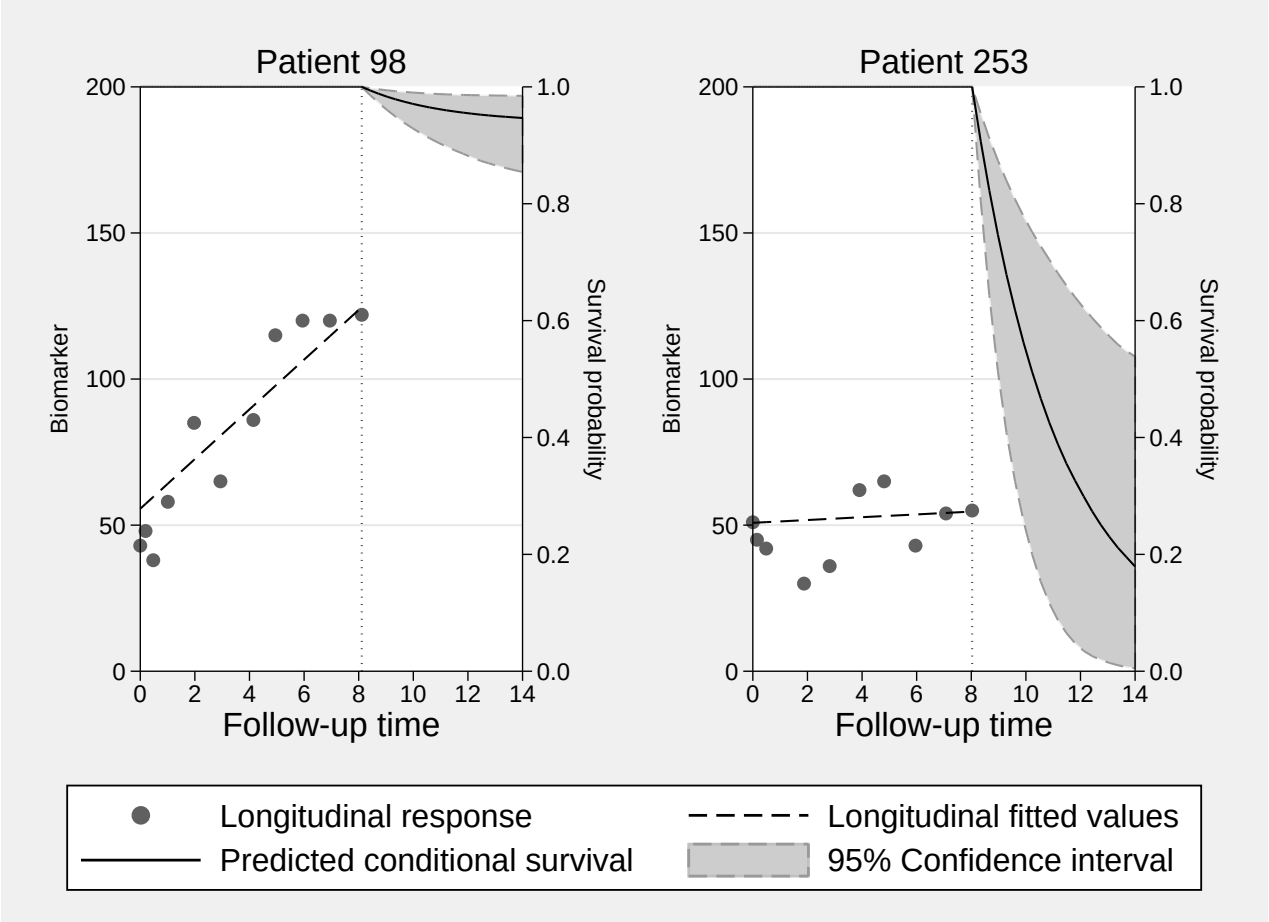
- More data $\rightarrow$ more questions
  - need for appropriate statistical modelling techniques, and implementations
- Growth in access to EHR
  - biomarkers $<$ patients $<$ GP practice area $<$ geographical regions...
- The standard challenges
  - time-dependent effects, non-linear covariate effects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

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
- bamlss 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?

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 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

(I think I made my life more difficult)

(some) Maths

For a one-level model with n response variables:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{\beta}) = \prod_{i=1}^{n} p_i(y_i|\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{\beta})$$

For a two-level model:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{\beta}) = \prod_{i=1}^{n} \prod_{j=1}^{t} p_i(y_{ij}|\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{\beta})$$

(some) Maths

The log likelihood is obtained by integrating out the unobserved random effects

$$ll(\boldsymbol{\beta}) = \log \int_{\mathcal{R}^r} p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{\beta}) \phi(\boldsymbol{b}|\Sigma_{\boldsymbol{b}}) \, \mathrm{d}\boldsymbol{b}$$

we assume $\phi()$ is the multivariate normal density for $\boldsymbol{b}$, with mean vector $\mathbf{0}$ and variance-covariance matrix $\Sigma_{\boldsymbol{b}}$. We have $\Sigma_{\boldsymbol{b}}$ becoming block diagonal with further levels, with a block for each level

(some) Maths

Alternatively, exploiting conditional independence amongst level $l-1$ units, given the random effects at higher levels,

$$ll(\boldsymbol{\beta}) = \log \int \phi(\boldsymbol{b}^{(L)}|\boldsymbol{\Sigma}^{(L)}) \prod p^{(L-1)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{b}^{L}, \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{b}^{(L)}$$

where, for $l = 2, \ldots, L$

$$p^{(l)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{B}^{l+1}, \boldsymbol{\beta}) = \int \phi(\boldsymbol{b}^{(l)}|\boldsymbol{\Sigma}^{(l)}) \prod p^{(l-1)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{B}^{l}, \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{b}^{(l)}$$

Estimation challenges

- At each level, we need to integrate out our normally distributed random effects
- Generally this is done using Gauss-Hermite numerical quadrature

  `intmethod(mvaghermite | ghermite)`
- Issue with GH quadrature is it doesn't scale up well:
  - 7-point quadrature; for 1 random effect we evaluate our function at 7 points
  - 7-point quadrature; for 6 random effects, we evaluate it at $7^6 = 117,649$ points

Estimation challenges - alternatives

- An alternative is Monte Carlo integration
- Also known for its use in maximum simulated likelihood - see the special issue in the Stata Journal Vol 6 No 2
- This is a rather brute force approach, but it's usefulness is in it's simplicity

$$L(\boldsymbol{\theta}) = \int f(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{b})\phi(\boldsymbol{b})\partial\boldsymbol{b} = \frac{1}{m}\sum_{u=1}^{m} f(y|\theta, \boldsymbol{b}_u)$$

The important thing to note is m doesn't have to change when extra random effects are added.

Estimation challenges - alternatives

Monte Carlo integration can be improved by:

- antithetic sampling
- Halton sequences
- an adaptive procedure just like adaptive GH quadrature, resulting in an importance sampling approximation

Extensions - level-specific random effect distributions

$$ll(\boldsymbol{\theta}) = \log \int \phi_L(\boldsymbol{b}^{(L)}|\boldsymbol{\Sigma}^{(L)}) \prod p^{(L-1)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{b}^L, \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{b}^{(L)}$$

where, for $l = 2, \ldots, L$

$$p^{(l)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{B}^{l+1}, \boldsymbol{\beta}) = \int \phi_l(\boldsymbol{b}^{(l)})|\boldsymbol{\Sigma}^{(l)} \prod p^{(l-1)}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{B}^l, \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{b}^{(l)}$$

Extensions - level-specific random effect distributions and integration techniques

- This formulation now allows us to specify different distributions at each level
- Assess robustness using the t-distribution
- Issue of which integration techniques to apply at each level
  - e.g. one random effect at level 1, many at level 2, then use AGHQ at level 3, and MCI at level 2

```
intmethod(mvaghermite mcarlo)
redistribution(normal t) df(3)
```

Standard linear predictor

The standard linear predictor for a general level model can be written as follows,

$$\eta = \boldsymbol{X}\boldsymbol{\beta} + \sum_{l=2}^{L} \boldsymbol{X}^l \boldsymbol{b}^l$$

where subscripts are omitted. We have $\boldsymbol{X}$ our vector of covariates, which could vary at any level, with associated fixed effect coefficient vector $\boldsymbol{\beta}$, and $\boldsymbol{X}^l$ the vector of covariates with random effects $\boldsymbol{b}^l$ at level l.

Extended linear predictor

$$\eta_i = g_i(E[y_i|\boldsymbol{X}, \boldsymbol{b}]) = \sum_{r=1}^{R_i} \prod_{s=1}^{S_{ir}} \psi_{irs}$$

where $g_i()$ is the link function for the ith outcome. To maintain generality, $\psi_{irs}(t)$ can take many forms, including,

$$\psi_{irs}(t) = X$$
$$\psi_{irs}(t) = \beta$$
$$\psi_{irs}(t) = b$$
$$\psi_{irs}(t) = q(t)$$
$$\psi_{irs}(t) = d_{rs}(E[y_j]), \qquad \text{where } j = 1, \ldots, k, j \neq i$$

Extended linear predictor

$$\eta_i = g_i(E[y_i|\boldsymbol{X}, \boldsymbol{b}]) = \sum_{r=1}^{R_i} \prod_{s=1}^{S_{ir}} \psi_{irs}$$

where $g_i()$ is the link function for the ith outcome. To maintain generality, $\psi_{irs}(t)$ can take many forms, including,

$$\psi_{irs}(t) = X$$
$$\psi_{irs}(t) = \beta$$
$$\psi_{irs}(t) = b$$
$$\psi_{irs}(t) = q(t)$$
$$\psi_{irs}(t) = d_{rs}(E[y_j]), \qquad \text{where } j = 1, \ldots, k, j \neq i$$

And who says each outcome model can only have one complex predictor?

ssc install merlin (Stata)

Title

merlin — Mixed effects regression for linear and non-linear models

Syntax

merlin *models* [*if*] [*in*] [, *options*]

where *models* are the model specifications; see **merlin models**.

options	Description
model_description_options	fully define, along with *models*, the model to be fit
estimation_options	method used to obtain estimation results, including specifying initial values
reporting_options	reporting of estimation results

Also see **merlin postestimation** for features available after estimation.

`merlin` in Stata

- Everything I've talked about is available in the `merlin` package in Stata
- `merlin` has many extensions, such as
  - Alternative models, such as spline based survival models
  - Extending sharing between outcomes, motivated by joint modelling
  - User-defined likelihood functions
  - Other things...

`merlin` in R

- is catching up (slowly)
- a first version is on CRAN

Distributional choices

- Gaussian, Poisson, binomial, beta, negative binomial, ordinal (logit or probit link)
- exponential, Weibull, Gompertz, log-normal, log-logistic, gamma, Royston-Parmar, splines on log hazard scale
- non-linear outcome models
- user-defined hazard functions
- many (many) more to add...

An example

- we're going to fit 15 models
- each of them is applied to the same dataset
- some of them can be considered *new* models
- we can fit all of them with a single line of code (ok, it may get quite long...)

- 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
- 1,945 measurements of serum bilirubin, among other things

data

id	time	logb	prothr~n	trt	stime	cancer	other
1	0	2.674149	12.2	D-penicil	1.09517	1	0
1	.525682	3.058707	11.2	D-penicil	.	.	.
2	0	.0953102	10.6	D-penicil	14.1523	0	1
2	.498302	-.2231435	11	D-penicil	.	.	.
2	.999343	0	11.6	D-penicil	.	.	.
2	2.10273	.6418539	10.6	D-penicil	.	.	.
2	4.90089	.9555114	11.3	D-penicil	.	.	.
2	5.88928	1.280934	11.5	D-penicil	.	.	.
2	6.88588	1.435084	.	D-penicil	.	.	.
2	7.8907	1.280934	.	D-penicil	.	.	.
2	8.83255	1.526056	.	D-penicil	.	.	.

a model

```
merlin (logb                                /// log serum bilirubin
               time                         /// covariate
               ,                            /// options
               family(gaussian)             /// distribution
       )
```

a model

```
merlin (logb                        /// log serum bilirubin
               time                 /// covariate
               time#trt             /// interaction
               ,                    /// options
               family(gaussian)     /// distribution
       )                            ///
```

a model

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

a model

```
merlin (logb                                /// log serum bilirubin
                time                        /// covariate
                time#trt                    /// interaction
                M1[id]@1                    /// random intercept
                time#M2[id]@1               /// random slope
                ,                           /// options
                family(gaussian)            /// distribution
        )
```

a model

```
merlin (logb                                /// log serum bilirubin
               time                         /// covariate
               time#trt                     /// interaction
               M1[id]@1                     /// random intercept
               time#M2[id]@1                /// random slope
               ,                            /// options
               family(gaussian)             /// distribution
       )                                    ///
       (pro                                 /// prothrombin index
               rcs(time, df(3))             /// covariate
               , family(gamma)              /// distribution
       )                                    ///
```

a model

```
merlin (logb                                /// log serum bilirubin
               time                         /// covariate
               time#trt                     /// interaction
               M1[id]@1                     /// random intercept
               time#M2[id]@1                /// random slope
               ,                            /// options
               family(gaussian)             /// distribution
       )                                    ///
       (pro                                 /// prothrombin index
               rcs(time, df(3))             /// covariate
               M3[id]@1                     /// random effect
               , family(gamma)              /// distribution
       )                                    ///
```

a model

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

a model

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

a model

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

a model

```
merlin (logb                                    /// log serum bilirubin
               time                             /// covariate
               time#trt                         /// interaction
               M1[id]@1                         /// random intercept
               time#M2[id]@1                    /// random slope
               ,                                /// options
               family(gaussian)                 /// distribution
       )                                        ///
       (pro                                     /// prothrombin index
               rcs(time, df(3))                 /// covariate
               M3[id]@1                         /// random effect
               , family(gamma)                  /// distribution
       )                                        ///
       (stime  trt                              /// response + covariate
               dEV[logb] EV[pro]                /// associations
               , family(rp, df(3)               /// distribution
                       failure(other))          /// event indicator
       )                                        ///
       ,                                        /// main options
       covariance(unstructured)                 /// vcv
       redistribution(t) df(5)                  //  re dist.
```

a model

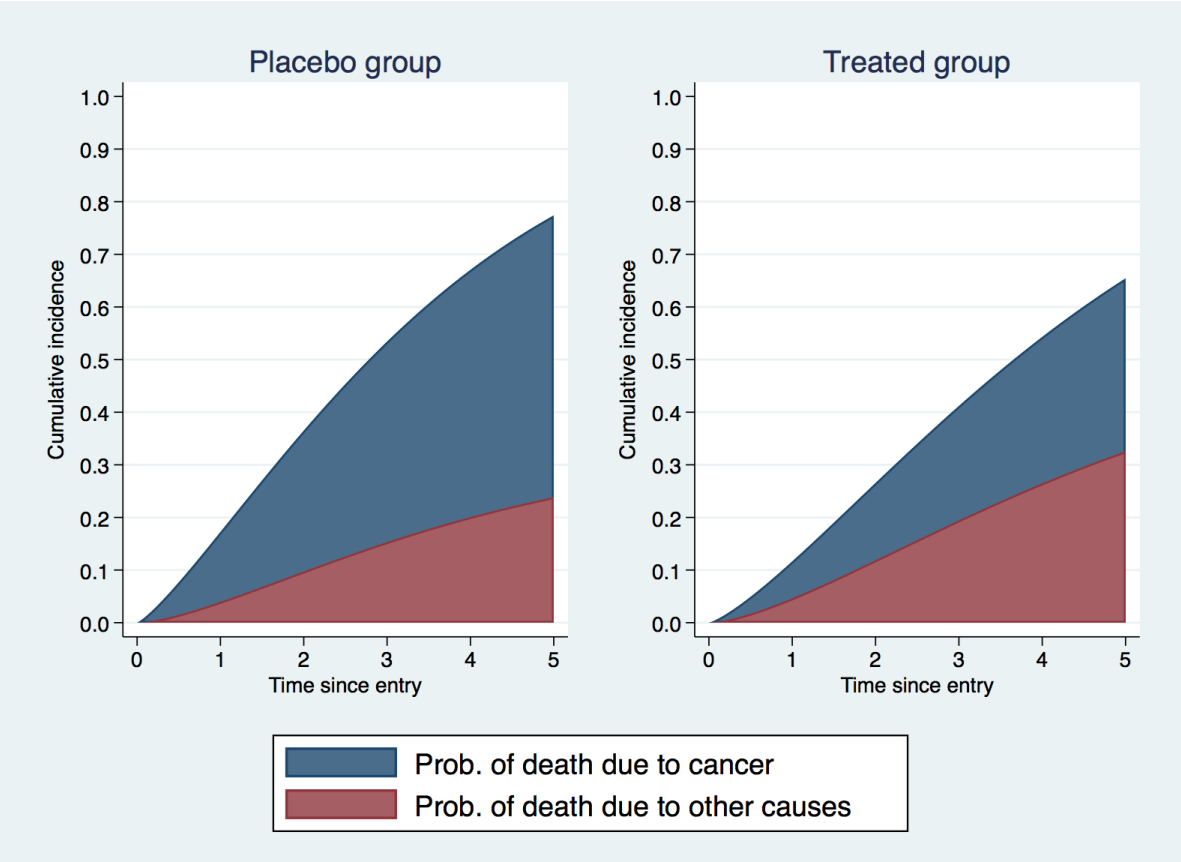
```
merlin (logb                                    /// log serum bilirubin
               time                             /// covariate
               time#trt                         /// interaction
               M1[id]@1                         /// random intercept
               time#M2[id]@1                    /// random slope
               ,                                /// options
               family(gaussian)                 /// distribution
       )                                        ///
       (pro                                     /// prothrombin index
               rcs(time, df(3))                 /// covariate
               M3[id]@1                         /// random effect
               , family(gamma)                  /// distribution
       )                                        ///
       (stime  trt                              /// response + covariate
               trt#fp(stime, power(0))          /// tde
               dEV[logb] EV[pro]                /// associations
               , family(rp, df(3)               /// distribution
                        failure(other))         /// event indicator
       )                                        ///
       ,                                        /// main options
       covariance(unstructured)                 /// vcv
       redistribution(t) df(5)                  //  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

```
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 non-linear model

```
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))
```

Things 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 wrapper functions to allow a simpler syntax for special cases
- `merlin`'s minions...
  - `stmixed` (`excalibur`) for multilevel survival analysis (now published in Stata Journal)
  - `predictms2` (`galahad`) - multi-state survival analysis
  - TBC (`lancelot`) - meta-analysis
  - TBC (`arthur`) - to be revealed next!

a surprise

Two useful features of `merlin` are:

- `EV[depvar/#]` element type
  - implemented for their use in joint longitudinal-survival models
- `family(null)`
  - implemented for use with user-defined models

their combination gives merlin some new capabilities

a 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?

a 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.

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

output options	Description
family(*fam_spec***)**	distributional family for the output/response, see **merlin families**
link(*type***)**	link function for the response model

a 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)
```

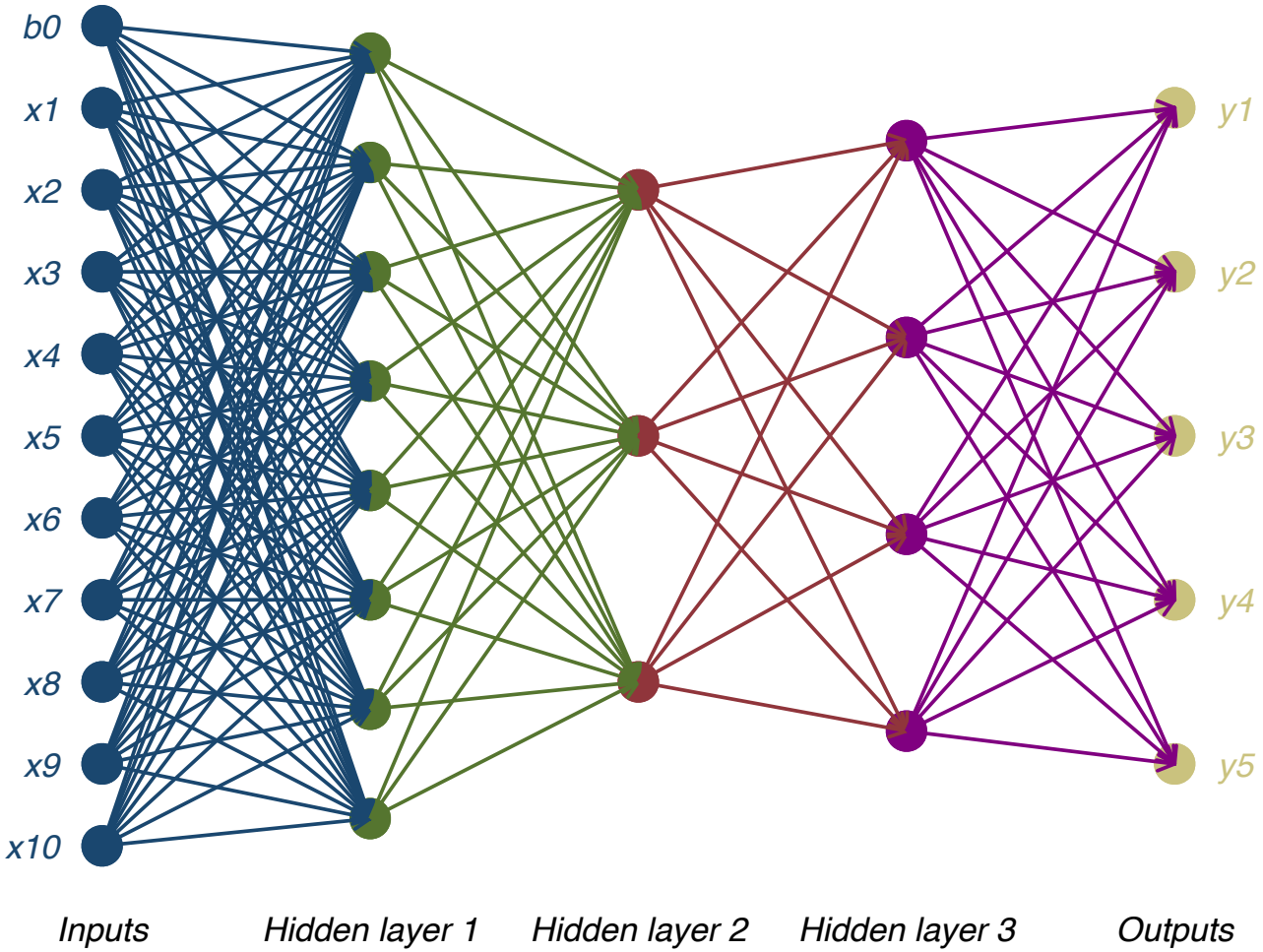
a surprise

```
merlin (y x1_nn x2_nn EV[4] EV[5] EV[6]
                              , 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)
```

```
. nnplot , inputs(10) outputs(5) hlayers(3) hnodes(8 3 4)
```

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
- Dynamic risk prediction - predictions are a key focus of the `merlin` engine
- It's general, and hence it can be slow(er)

www.mjcrowther.co.uk/software/merlin

Papers

- Crowther MJ. Extended multivariate generalised linear and non-linear mixed effects models. https://arxiv.org/abs/1710.02223
- Crowther MJ. merlin - a unified framework for data analysis and methods development in Stata. *Stata Journal* (To appear). (Pre-print: https://arxiv.org/abs/1806.01615)
- Crowther MJ. Multilevel mixed effects parametric survival analysis: Estimation, simulation and application. *Stata Journal* 2019;19(4):931-949. (Pre-print: https://arxiv.org/abs/1709.06633)
- Gasparini A, Abrams KR, Barrett JK, Major JK, Sweeting MJ, Brunskill NJ, Crowther MJ. Mixed effects models for healthcare longitudinal data with an informative visiting process: a Monte Carlo simulation study. *Statistica Neerlandica* 2020;74(1):5-23. (Pre-print: