A Note on Estimating a Competing Risk Model with Transition Specific Unobserved Heterogeneity

Simen Gaure
The Frisch Centre for Economic Research
Scientific Computing Group, USIT, University of Oslo

14/12-2006

1 Introduction

This note is about estimating fairly general competing risk models. It describes the ideas and functionality of an estimation program in production use at the Frisch Centre.

Say we have N individuals who may be in S different states. From each state they may make one of T different transits and possibly end up in a different state.

A typical example is from labour market modelling, where unemployed individuals may transit to either a labour market programme or to job. An indivdual on a labour market programme (with exogeneously determined length) may transit only to a job. In this example the states are "unemployed" and "on programme" whereas the transits are "to job" and "to programme". For an individual in state "on programme", the only legal transit is "to job". I.e. there is a single risk for these.

We will use this example now and then, though of course the estimation program works equally well if e.g. the individuals are cancer patients and the transitions are "to medical treatment A", "to cured" and "to the graveyard".

Associated with a transition t is an individual *hazard* rate θ which depends on some covariates X_i (with $i = 1..N$). We model this hazard as

$$\theta_i = e^{X_i \cdot \beta}$$

(where β is a parameter vector and $\cdot$ is the inner product).

We may use a different set of covariates for different transitions. We want to estimate the vector β with the maximum likelihood estimator, but there are some further complications.

To compute the likelihood $L_i(X_i, \beta)$ of a particular individual transit we must integrate the survival rate over the period in the current state and multiply with the hazard rate at the end of the period. The form of the integrated survival depends on how we model the duration time. In particular we may use continuous duration time, or discrete duration time.

Some of the covariates X_i may depend on the duration that has been spent in the state, this means that to avoid bias in β we must also model *unobserved heterogeneity*. Following Lindsay, and Heckman and Singer, we use a discrete heterogeneity distribution and we allow for transition specific heterogeneity. I.e. we have probabilities (masspoints) $p = (p_i)_{i=1}^K$ with $\sum p_i = 1$ and vectors $v = (v_t)_{t=1}^T$ of length K for each transition t. I.e. so that v_{tj} is the heterogeneous intercept for masspoint j for transition t. Let also $v_{\cdot j}$ denote the other projection, i.e. the location vector of length T: $v_{\cdot j} = (v_{1j}, v_{2j}, \ldots, v_{Tj})$.

These things enter the hazard as

$$\theta_{itj} = e^{X_i \cdot \beta + v_{tj}}$$

and we have to integrate both the survival rate over the period, and the heterogeneity over the heterogeneity distribution, so that our individual likelihood becomes

$$L(X_i, \beta, K, v, p) = \sum_{j=1}^{K} p_j \Theta(X_i, \beta, v_{\cdot j}).$$

Where Θ is the likelihood conditional on a particular masspoint.

The v_{tj} enters as a *random intercept*, so we have no implicit intercept in β.

Note that there are no other functional forms in our likelihood, in particular even with continuous duration time, the hazard rate is considered to be piecewise constant, though with completely flexible interval lengths.

2 Overall strategy

Our individual likelihoods from the previous section must be multiplied over the individuals to get a function of the parameters β and our heterogeneity distribution. We take the logarithm to get the following log-likelihood

$$\mathcal{L}(\beta, K, v, p) = \sum_{i=1}^{N} \log L(X_i, \beta, K, v, p).$$

Our task is to maximize this function with respect to all its arguments subject to the constraint $K \leq N$ (this is from Lindsay). Fortunately, Heckman and Singer has taken care of some of it. We may handle K in the following way.

1. Start with $K = 1$, maximize $\mathcal{L}(\beta, K, v, p)$ w.r.t. (β, v, p).
2. Now, increase K by 1, keep β fixed and search for a *better* likelihood $\mathcal{L}(\beta, K, v, p)$ by varying (v, p).
3. Do a maximization of $\mathcal{L}(\beta, K, v, p)$ w.r.t (β, v, p), using (β, v, p) from the previous step as an initial value. If the new likelihood is an *improvement* go to step 2, otherwise terminate.

Some words are emphasized above, in step 2, we need to specify what we mean by *better*. In step 3 we need to specify what we mean by *improvement*. We also need to decide how to search in step 2, and how to maximize in step 3. When we have finished, we need to decide which model (i.e. which K) to use.

Note that the algorithm given here is slightly different from Heckman and Singer's. In step 2, we could terminate the algorithm if we couldn't improve the likelihood (in the sense that a certain directional derivative can't be made positive), but this is conditional on the fixed β. There is a chance that step 3 may actually find a better likelihood (by changing β slightly), even if step 2 can't do it.

2.1 How we search

In step 2 we need to search for a new masspoint. Following Heckman and Singer we do this as follows:

Let $p = (p_1, \ldots, p_K)$ and $v = (v_{tk})$ be the old probabilities and location points. Let $p_{K+1} = 0$ and let w be a T-dimensional vector. Let

$$\mathcal{M}(w, \rho) = \mathcal{L}(\beta, K+1, [vw], ((1-\rho)p, \rho))$$

where $[vw]$ is the matrix v extended (by concatenation) with the vector w. Form the directional derivative

$$G(w) = \lim_{\rho \to 0} \frac{\mathcal{M}(w, \rho) - \mathcal{M}(w, 0)}{\rho}.$$

We must search for a T-dimensional vector w which makes $G(w)$ positive. There are several ways to do this, e.g. local maximization, grid search, adaptive grid search, etc. We have picked *simulated annealing* as implemented by Goffe et al. For our purposes we have usually limited each coordinate v to the interval $[-5, 1]$.

As we get more points in our heterogeneity distribution, we replace the fixed search inverval by $[E(v) - 4\sigma(v), E(v) + 4\sigma(v)]$.

Note that we do not maximize $G(w)$, we stop as soon as $G(w)$ becomes positive. Once we have found such a w we use it as the new location point $v_{\cdot K+1}$, we set $\rho = 10^{-4}$ and use $((1-\rho)p, \rho)$ as the new probabilities.

If we do not find a w with $G(w) > 0$, we use the best one, i.e. we have done a global maximization of $G(w)$ and use the w which maximizes $G(w)$.

2.2 How we maximize

The maximization in step 1 and 3, being likelihood maximization, should really be a global maximization. With $N \approx 10^6$, $T = 7$, $K \approx 50$ and $\beta \in \mathbb{R}^{3000}$ this is beyond our current computational capabilities. We have resorted to local maximization, more specifically a combination of BFGS and Fisher scoring (i.e. Newton's method with the Hessian replaced by the Fisher matrix).

Moreover, we have splitted the maximization into two parts. We have seen that often the maximization leads to quite small changes in β, so we first maximize only with respect to (v, p), then w.r.t. (β, v, p).

We typically combine BFGS and Fisher scoring as follows:

a. Set b to 50. Set n to 30.

b. Do up to b iterations of BFGS.

c. Do up to n Newton iterations. If the resulting gradient is smaller than 10^{-9}, terminate the algorithm with success. If $n \geq 100$, then terminate with failure, otherwise increase b by 100, n by 10 and go to step b.

It turns out that both BFGS and Newton may run into problems, but the above combination often gets around problematic regions. The number of iterations and the convergence criterion may be adjusted.

2.3 When to stop, what to do

If the log-likelihood does not improve by at least 0.01 we do not consider it an improvement.

When our algorithm has terminated there are several model candidates. We may pick the one with the highest likelihood, we may pick the one with the highest AIC. If using AIC as a criterion it would be tempting to terminate the algorithm when the AIC does not improve, but we don't recommend that. The reason is that the AIC may improve in later steps.

2.4 Further complications

Sometimes, during maximization, the heterogeneity distribution ends up as degenerate. I.e. one or more probabilities are estimated as zero, or some location vectors are equal (or close enough to be considered equal). In this case, we remove the superfluous point(s) from the distribution and re-estimate.

To this end, a probability of less than 10^{-6} is considered to be zero. Two location vectors $v_{\cdot i}$ and $v_{\cdot j}$ are considered to be equal if $\|v_{\cdot i} - v_{\cdot j}\|_\infty < 0.1$.

There are more complications. In our example, some people never enter a programme, their only transition is to "employed". This means that technically, the risk that they enter a programme is zero, i.e. a defective risk. The maximization algorithm may sometimes estimate a very large negative value in this case. E.g. v_{26} is very negative. Then $e^{v_{26}}$ is *numerically* equal to zero and we are not able to compute meaningful gradients, nor refine v_{26}. The Fisher matrix becomes degenerate, and we can't invert it to obtain standard errors for any parameters. In this case we *mark* v_{26} as $-\infty$ and leave it as a constant in the estimation.

Our complications do far from end here. In particular for discrete durations we have occasional numerical problems. I.e. we have expressions like $1 - e^{-e^x}$ and their derivatives. For certain arguments we then use a Taylor approximation to alleviate numerical difficulties. Also, for some expressions it's more accurate to work directly with their logarithms.

There's also a trick which has proved to be very useful. During estimation, we sometimes encounter values of parameters which makes the likelihood of some individuals very close to zero, so that their logarithm becomes very negative. It will typically be hard to compute the gradient with any reasonable accuracy for these individuals. We have solved this problem by setting the log-likelihood for these individuals to a large negative number, whereas their gradient is set to zero. This introduces a discontinuity in the gradient, but it turns out that the maximization algorithm most often moves away from such regions, relying on the gradient contribution from the "sane" individuals.

3 Duration dependence and *implicit dummies*

We have briefly mentioned duration dependence. I.e. the hazard rate may depend on how long the individual has been in the current state. In the literature one sees various functional forms of duration dependence. One popular form has been the Weibull distribution, and there are other forms, decreasing, increasing and with various regular humps and bumps.

We have used no such functional form. We depend on estimating duration dependence with dummy variables. E.g. for a dataset with monthly data with durations up to 5 years, we use 60 dummies, one for each month, to estimate duration dependence. In some applications we get collections of over 100 such dummies. Therefore we have optimized particularly for long strings of dummies. The user may e.g. specify an ordinal variable "dur" as *implicit dummy* from 1 to 60 with reference 13. "dur" should then have integer values between 1 and 60 and the estimation program will create 59 dummies ("dur1" – "dur60", except the reference "dur13") corresponding to each value of "dur". This is not merely a convenience for the user, the fact that no more than 1 of the 59 dummies is non-zero at any time, is actively used to speed up the execution, effectively treating the string of dummies as one variable (which it really is), both with respect to speed and space.

Since we have this flexible (though piecewise constant) duration dependence, it's possible to track duration dependence which e.g. stems from pure policy decisions, e.g. we may see bumps prior to (partial) benefit exhaustion at 18 and 36 months. Implicit dummies may of course also be used to track calendar time dependence, e.g. the business cycle.

Note that the duration dummies will enter the likelihood just like any other covariate, this means that we use a model which is known as "proportional hazard", or, taking into account the heterogeneity distribution, "mixed proportional hazard".

Note also that with this many parameters we require a lot of data to get sensible standard errors. Since we typically use these methods for analyzing register data this is not a big problem for us.

4 Random Coefficients

Just like we modeled the heterogeneity with a discrete distribution, we may estimate random coefficients. From above, our individual hazard

$$\theta_{itj} = e^{X_i \cdot \beta + v_{tj}}$$

is replaced by

$$\theta_{itj} = e^{X_i \cdot \beta + \alpha_j r_i + v_{tj}}$$

where r_i is some covariate and α_j is a *random coefficient*. Note that the unobserved heterogeneity is merely a special case of a random coefficient corresponding to a covariate which is constantly equal to 1.

5 Linear Predictors

When unemployed people get a job, it's interesting to analyze how good a job it is, e.g. by looking at the wage. We update our likelihood to include a linear predictor for the (log)wage e.g. like

$$Y_i = X_i \cdot \gamma + v + \mathcal{N}(0, \sigma)$$

where Y_i is a covariate (the (log)wage), X_i are covariates (possibly different from the ones occuring in the hazard), γ is a set of parameters to estimate, v is heterogeneity and $\mathcal{N}(0, \sigma)$ is an error term normalized to have mean 0. The individual likelihood is

$$\phi_{ij} = N_\sigma(Y_i - (X_i \cdot \gamma + v_j))$$

where

$$N_\sigma(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

is the normal density with mean 0. This likelihood ϕ_{ij} is multiplied with the hazard likelihood Θ in 1. We get parameters (γ, σ, v_j) to maximize in addition to the ones from the hazard.

Interestingely, we have seen that with a linear predictor we benefit from changing our maximization algorithm slightly. We still do the same search for a new point, but we do not do a separate maximization over the distribution (as mentioned in 2.2), instead we jump right to a maximization of all the parameters. This has, in our datasets, prevented the algorithm from repeatedly converging into a degenerate pit.

6 The integrated hazard

As mentioned above, we can either use continuous time, in which the duration until transit is exactly recorded, or discrete time where time is divided into intervals of equal length and we merely record in which interval the transit occurs.

6.1 Discrete time

We organize our data so that all covariates are constant within a unit time period. With time-varying covariates we therefore may have several "observations" with no transit, before the observation with the transit.

With discrete time we need to compute the probability that the observed outcome occured within the observed interval. The probability P_s of survival through a period is the product integral of the hazard:

$$P_s = \exp(-\sum_{t=1..T} \theta_t)$$

where θ_t is the hazard of doing transition t. In case the individual did not make a transition within a period, P_s will be the likelihood for this period.

In the case that a t-transition was made, the likelihood will be

$$P_t = (1 - P_s)\frac{\theta_t}{\sum_u \theta_u}.$$

I.e. the probability of non-survival times the probability of t-transition conditional that a transition was made.

6.2 Continuous time

Here we also organize our data into periods with constant covariates. The periods may be of arbitrary and different lengths. The likelihood for a period of length ℓ can be computed as: Survival:

$$P_s = \exp(-\ell \sum_t \theta_t)$$

and t-transition:

$$P_t = P_s \theta_t$$

This is a density, rather than a probability.

7 Implementation

The estimation as described above is quite time-consuming. Although it is possible to write down the likelihood and feed it to some general purpose statistical package, this has not proven useful for the datasets we have. We have resorted to write the procedure in the de facto standard language for high performance computations, FORTRAN 90. We have used MPI to do the computations in parallel (over the dataset).

Since we want to have an easy way to specify a particular problem, we have written a preprocessor in perl, which takes a human-writeable specification and transforms it into problem-specific fortran code which is included in the estimation program. This has the additional benefit that many loop-limits in the program are compile-time constants, this aids optimization.

Since Fortran hardly can be said to be a rational language for data-manipulation, we have also written a post-processing procedure in R. This takes care of computing various statistics from the Fisher matrix.

In addition we have written a web-frontend in PHP to submit problems to the Titan cluster at the University of Oslo. (A cluster which the Frisch Centre has a share in).

7.1 Speed

As mentioned above, the estimation procedure is time-consuming, in particular with large datasets and large sets of parameters. In addition to coarse grain parallellism and implicit dummies, we have done quite detailed loop-level optimization. E.g. we try to ensure that all time-consuming loops are executed in the right order (forwards in memory) and that cache-lines are fully utilized before wasted. But, the single most important thing is to use a good blas and lapack. This also holds for the R post-processing routine.

As an example of this, our Newton step uses the Fisher matrix, which can be computed from the outer product of the individual gradients ∇L_i as

$$\sum_i (\nabla L_i)(\nabla L_i)'.$$

Doing this without further thought will be next to disastrous when the vector length becomes long (i.e. so long that the Fisher matrix no longer fits in the cache). To speed up this critical process we collect ∇L_i for 32–128 individuals at a time (depending on the machine architecture), and compute their contribution to the Fisher matrix with the blas level 3 routine “dsyrk”. With a vector length of 3000, the speed difference between a good and a poor blas may be an order of magnitude or more.