The Stata Journal (*yyyy*) *vv*, Number *ii*, pp. 1–13

Estimating survival functions after `stcox` with time-varying coefficients

Constantin Ruhe
Department of Politics and Public Administration
University of Konstanz
Konstanz, Germany
Constantin.Ruhe@uni-konstanz.de

Abstract. In many applications of the Cox model the proportional hazards assumption is not plausible. In these cases, the solution to non-proportional hazards usually consists of modeling the effect of the variable of interest as well as the interaction effect of this variable with some function of time. Although Stata provides a handy command to implement this interaction in `stcox`, it does not allow the typical visualizations using `stcurve` if `stcox` was estimated with the `tvc()` option. In this article, I provide a short workaround which enables to estimate the survival function after `stcox` with time-dependent coefficients. I introduce and describe a new program `scurve_tvc` which automates this procedure and which allows users to easily visualize survival functions for models with time-varying effects.

Keywords: st0001, `scurve_tvc`, `stcox`, `tvc()`, `stcurve`, `sts generate`, Cox model, proportional hazards, time-varying coefficients, survival function

1 Overview

In many disciplines, the time-varying effects of variables in duration analyses are of substantive interest (Box-Steffensmeier et al. 2003; Giolo et al. 2012; Nilsson and Nivre 2013). Assume that a cancer treatment x had short-term benefits, but long-term treatment would cause side-effects which eventually deteriorate a patients health situation. In this scenario, we would like to know if the long-term side-effects outweigh the short-term benefits. This would be the case if after a certain time, fewer people which received the treatment had survived compared to patients who did not receive treatment x. The comparison of these groups could be achieved with the respective survivor functions for each group (Putter et al. 2005). Calculating these estimates requires a short workaround, since Stata does not provide a build-in solution to plot the survival function with time-varying coefficients. I provide a solution below.

In the case that all covariates have constant effects, it is straightforward to calculate the survivor function for different scenarios, based on the estimated coefficients as well as the baseline survival functions (Kalbfleisch and Prentice 2002; Cleves et al. 2010). Let $h_0(t)$ be the baseline hazard function. In the Cox model, the hazard function for an individual i with covariates x is then asserted to be

$$h(t|x_i) = h_0(t)exp(x_i\beta) \tag{1}$$

From this we can calculate the cumulative hazard function:

$$H(t|x_i) = \int_0^t h(u|x_i)\,\mathrm{d}u = exp(x_i\beta)\int_0^t h_0(u)\,\mathrm{d}u = exp(x_i\beta)H_0(t) \tag{2}$$

Since the survivor function can be calculated from the cumulative hazard function, we get:

$$S(t|x_i) = exp(-exp(x_i\beta)H_0(t)) = S_0(t)^{exp(x_i\beta)} \tag{3}$$

In the case of time-varying effects, this calculation becomes more complicated. To allow for a changing effect, an interaction of the variable with some function of time can be included as a time-varying covariate. If the function $f(t)$ by which the effect varies with time is known, the effect can easily be modeled. Consider a variable z with such a time-varying effect. If we enter the interaction of z with time as a time-varying covariate, the hazard can be written as

$$h(t|z_i) = h_0(t)exp(z_i\gamma + z_i f(t)\delta) = h_0(t)exp(z_i[\gamma + f(t)\delta]) = h_0(t)exp(z_i\beta_t) \tag{4}$$

While this easily accounts for the time-varying effect, the calculation of the survival function becomes more complicated. Once the predictor variables in the model are interacted with time, the linear combination of predictors $z_i\beta_t$ depends on time and remains in the integral in (2).

Below, I provide an example how to estimate these survivor functions in Stata. Thereafter, I describe the new program `scurve_tvc` which automates this procedure.

2 Estimating survival functions with tvc()

Consider a case with a binary treatment variable x and a binary confounder called *control*, whereby the effect of x changes over time. The following code generates duration data for this setting, whereby the observations are censored after 30 observations (cf. Crowther and Lambert 2012). The data come from the following exponential duration model:

$$h(t|x_i) = exp(ln(0.05) - 0.9x_i + 0.6ln(t)x_i + 1.5control_i)$$

```
. clear
. set seed 94215841
. set obs 1000
number of observations (_N) was 0, now 1,000
.
. gen x=(runiform()>.5)
```

```
. gen control=(runiform()>.5)
.
. survsim ftime, cov(x -.9 control 1.5) tde(x .6) distribution(exponential) ///
>         lambda(.05)
. gen failure=(ftime<=30)
. replace ftime=30 if ftime>30
(65 real changes made)
.
. stset ftime, failure(failure)
     failure event:  failure != 0 & failure < .
obs. time interval:  (0, ftime]
 exit on or before:  failure
```

1000	total observations
0	exclusions

1000	observations remaining, representing
935	failures in single-record/single-failure data
9693.849	total analysis time at risk and under observation
	at risk from t = 0
	earliest observed entry t = 0
	last observed exit t = 30

To model the time-dependent effect, users of Stata could draw on `stcox` and the `tvc()` option. Unfortunately, however, Stata is unable to estimate survival functions in the presence of time-dependent effects.

```
. stcox x control, tvc(x) texp(ln(_t)) nohr nolog
        failure _d:  failure
  analysis time _t:  ftime

Cox regression -- no ties

No. of subjects =        1,000                 Number of obs    =       1,000
No. of failures =          935
Time at risk    =  9693.849402
                                               LR chi2(3)       =      468.86
Log likelihood  =   -5468.3546                 Prob > chi2      =      0.0000
```

_t	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
main						
x	-.931918	.132382	-7.04	0.000	-1.191382	-.672454
control	1.464325	.0768318	19.06	0.000	1.313737	1.614912
tvc						
x	.6156235	.065858	9.35	0.000	.4865441	.7447028

```
Note: Variables in tvc equation interacted with ln(_t).
.
. capture noisily stcurve, survival at(x=1) at(x=0)
this post-estimation command is not allowed after estimation with tvc();
see tvc note for an alternative to the tvc() option
```

In a randomized clinical trial, the results for both groups are easily compared using e.g. the Kaplan-Meier survival estimates. However, if our data stems from observational studies and requires us to adjust for a larger number of covariates, this comparison is no longer useful. Moreover, we might want to predict the survival function for different subgroups to assess the relative success of an intervention in different contexts (cf. Putter et al. 2005). In this case, a manual workaround can help us to get the intuition how the two groups evolve over time as well as how the effect differs across subgroups in our sample. Below, I outline this workaround using the example data.

First, we need to estimate the time varying coefficient manually:

```
. gen id=_n
. stset ftime, id(id) failure(failure)
                id:  id
     failure event:  failure != 0 & failure < .
obs. time interval:  (ftime[_n-1], ftime]
 exit on or before:  failure

------------------------------------------------------------------------------
       1000  total observations
          0  exclusions
------------------------------------------------------------------------------
       1000  observations remaining, representing
       1000  subjects
        935  failures in single-failure-per-subject data
   9693.849  total analysis time at risk and under observation
                                             at risk from t =         0
                                  earliest observed entry t =         0
                                       last observed exit t =        30

.
. stsplit, at(failures)
(935 failure times)
(497,420 observations (episodes) created)
.
. generate x_t = x*ln(_t)
.
. stcox x x_t control, nolog nohr
         failure _d:  failure
   analysis time _t:  ftime
                 id:  id

Cox regression -- no ties

No. of subjects =        1,000                 Number of obs    =     498,420
No. of failures =          935
Time at risk    =  9693.849402
                                               LR chi2(3)       =      468.86
Log likelihood  =   -5468.3546                 Prob > chi2      =      0.0000

------------------------------------------------------------------------------
          _t |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   -.931918    .132382    -7.04   0.000    -1.191382    -.672454
         x_t |    .6156235   .065858     9.35   0.000     .4865441    .7447028
     control |    1.464325   .0768318    19.06   0.000     1.313737    1.614912
------------------------------------------------------------------------------
```

We could now use the `stcurve` command, but setting *x_t* to some value would ignore the fact that this variable is not constant, but rather varies with time. Hence, we need to proceed with the manual workaround and grab the stored coefficient estimates:

```
. matrix b=e(b)
```

We now use `sts generate` (or alternatively `predict` with the option `basehc`) to store the estimated hazard component, $\Delta H(t_j) = H(t_j) - H(t_{j-1})$, adjusting all variables to zero. Smoothing the variable *baseline* would generate an estimate of the baseline hazard function. The cumulative function of the variable will give us an estimate of the baseline cumulative hazard function.

```
. sts generate baseline=h, adjust(x x_t control)
```

Here, however, we are interested in a comparisson of the survival function for four different scenarios: *x=1* and *control=0*, *x=0* and *control=0*, *x=1* and *control=1*, *x=0* and *control=1*. Since the effect of *x* varies with time, we cannot simply calculate the function from the baseline cumulative hazard or survival function. We therefore calculate the scenario-specific hazard contribution based on Kalbfleisch and Prentice (2002, 114ff.).

$$\Delta H(t_j|x_i) = [1 - (1 - \Delta H(t_j))]^{exp(x_i\beta)} \tag{5}$$

By summing up these values, we approximate the cumulative hazard function for each scenario. From this, we can calculate the estimated survival function.

$$S(t_j|x_i) = exp[-\sum \Delta H(t_j|x_i)] \tag{6}$$

The workaround implements these steps as follows:

```
.
. preserve
.
. sort _t
. collapse (mean) baseline, by(_t)
.
.
. *scenario 1: x=1 and control=0
. gen b_x_nocontrol = 1-(1-baseline)^exp(b[1,1]+b[1,2]*ln(_t))
(1 missing value generated)
. gen H_x_nocontrol = sum(b_x_nocontrol)
. gen S_x_nocontrol = exp(-H_x_nocontrol)
.
.
. *scenario 2: x=0 and control=0
. gen b_nox_nocontrol = baseline
(1 missing value generated)
. gen H_nox_nocontrol = sum(b_nox_nocontrol)
. gen S_nox_nocontrol = exp(-H_nox_nocontrol)
```

```
.
.
. *scenario 3: x=1 and control=1
. gen b_x_control = 1-(1-baseline)^exp(b[1,1]+b[1,2]*ln(_t)+b[1,3])
(1 missing value generated)
. gen H_x_control = sum(b_x_control)
. gen S_x_control = exp(-H_x_control)
.
.
. *scenario 4: x=0 and control=1
. gen b_nox_control = 1-(1-baseline)^exp(b[1,3])
(1 missing value generated)
. gen H_nox_control = sum(b_nox_control)
. gen S_nox_control = exp(-H_nox_control)
.
.
. keep S_x_nocontrol S_nox_nocontrol S_x_control S_nox_control  _t
. rename _t _tx
. save my_surv_curve, replace
(note: file my_surv_curve.dta not found)
file my_surv_curve.dta saved
.
. restore
.
. merge using my_surv_curve
(note: you are using old merge syntax; see [D] merge for new syntax)
.
```

This data can now be merged with the original data and plotted using simple line plots. Figure 1 compares the workaround estimates against Kaplan-Meier estimates for each scenario. The individual graphs were combined with the user-written program `grc1leg`. The graph demonstrates that the estimated survival functions are in reasonable agreement with the non-parametric Kaplan-Meier estimates for each scenario.

```
. sts graph if control==0, by(x) plot1opt(lpat(dash)) plot2opt(lpat(solid)) ///
>         scheme(sj) saving(km1, replace) ylab(0(.2)1, format(%9.1g)) title("") ///
>         subtitle("Control=0")
         failure _d:  failure
   analysis time _t:  ftime
                 id:  id
(note: file km1.gph not found)
(file km1.gph saved)
. sts graph if control==1, by(x) plot1opt(lpat(dash)) plot2opt(lpat(solid)) ///
>         scheme(sj) legend(off) saving(km2, replace) ylab(0(.2)1, format(%9.1g)) ///
>         title("") t1title("Control=1")
         failure _d:  failure
   analysis time _t:  ftime
                 id:  id
(note: file km2.gph not found)
(file km2.gph saved)
. line S_nox_nocontrol S_x_nocontrol _tx, c(J J) sort lp(dash solid) ///
>         scheme(sj) legend(off) saving(w1, replace) ylab(0(.2)1) ///
>         t1title("Control=0") xtitle("analysis time")
```

```
(note: file w1.gph not found)
(file w1.gph saved)
. line S_nox_control S_x_control _tx, c(J J) sort lp(dash solid) ///
>         scheme(sj) legend(off) saving(w2, replace) ylab(0(.2)1) ///
>         t1title("Control=1") xtitle("analysis time")
(note: file w2.gph not found)
(file w2.gph saved)
. grc1leg km1.gph km2.gph, leg(km1.gph) scheme(sj) ycommon ///
>         saving(km, replace) subtitle("Kaplan-Meier estimates")
(note: file km.gph not found)
(file km.gph saved)
. grc1leg w1.gph w2.gph, scheme(sj) ycommon saving(w, replace) ///
>         subtitle("Cox estimates")
(note: file w.gph not found)
(file w.gph saved)
. grc1leg km.gph w.gph, scheme(sj) ycommon leg(km.gph) col(1)
```

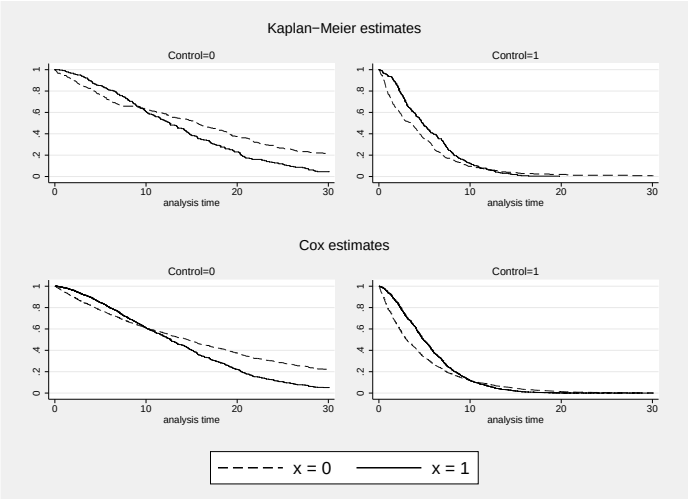

Figure 1: Comparing Kaplan-Meier survival estimates against estimated survivor function using the workaround.

3 The program `scurve_tvc`

The procedure is automated in the ado-file `scurve_tvc`. The program estimates a Cox model with time-varying coefficients from which it estimates, saves and plots the estimated survival function for as user-specified scenario. I describe the syntax, options and an illustrative example below.

3.1 Syntax

scurve_tvc [*if*] [*in*] , generate(*newvarname*)
at(*varname* # [*varname* # ...]) tvc(*varlist*) texp(*string*) [replace
ties(*string*) shared(*varname*) strata(*varname*) graph plotopts(*string*)]

3.2 Description

scurve_tvc fits stcox with time-varying effects and calculates the survival curve for specific covariate values.

generate(*newvarname*) creates the variable *newvarname*to store the estimated survival curve. If strata() is specified, scurve_tvc creates one variable for each stratum. The corresponding analysis time variable, which allows to plot the results, is saved in a new variable called *_tscurve*.

at(*varname* # [*varname* # ...]) specifies the covariates included in the model and the values for which the survival curve should be calculated.

tvc(*varlist*) specifies the covariates with time-varying coefficients. The variables in tvc() must also appear in at(). scurve_tvc will automatically stsplit the data at failure times to ensure a correctly estimated model. See help tvc note for more information.

texp(*string*) specifies the function of analysis time according to which the effect varies with time. For example, specifying texp(ln(_t)) would cause the variables with time-varying coefficients to be multiplied by the logarithm of analysis time.

3.3 Options

replace existing variable(s) with the new estimates.

ties(*string*) specifies the option how stcox handles tied failure times. See help stcox for details.

shared(*varname*) specifies a shared-frailty ID variable. See help stcox for details.

strata(*varname*) specifies a strata ID variables. See help stcox for details.

graph plots the predicted survival curve. If strata() is specified, the survival estimates for each stratum will be plotted.

plotopts(*string*) enables to customize the plot using options allowed with twoway line.

3.4 Example

Similar to the example above, consider a case with a binary treatment variable x, a binary confounder called *control1* and a continuous confounder called *control2*, whereby the effect of x changes over time. The following code generates duration data for this setting, whereby the observations are censored after 30 observations (cf. Crowther and Lambert 2012). The data come from the following Weibull duration model:

$$h(t|x_i) = 1.3t^{1.3-1}exp(ln(0.05) - 0.9x_i + 0.6ln(t)x_i - 1.3control1_i - 0.4control2_i)$$

```
. clear
. set seed 581265456
. set obs 1000
number of observations (_N) was 0, now 1,000
.
. gen x=(runiform()>.5)
. gen control1=(runiform()>.5)
. gen control2=rnormal()
.
. survsim ftime, cov(x -.9 control1 -1.3 control2 -.4) tde(x .6) ///
>         distribution(weibull) lambda(.05) gamma(1.3)
. gen failure=(ftime<=30)
. replace ftime=30 if ftime>30
(109 real changes made)
.
. gen id=_n
.
. stset ftime, id(id) failure(failure)
               id:  id
    failure event:  failure != 0 & failure < .
obs. time interval:  (ftime[_n-1], ftime]
 exit on or before:  failure
------------------------------------------------------------------------------
       1000  total observations
          0  exclusions
------------------------------------------------------------------------------
       1000  observations remaining, representing
       1000  subjects
        891  failures in single-failure-per-subject data
  13122.843  total analysis time at risk and under observation
                                                at risk from t =         0
                                     earliest observed entry t =         0
                                          last observed exit t =        30
```

Assume that x is a cancer treatment from which we know that it causes severe long-term side-effects. Hence, we would want to know under which circumstances the treatment is beneficial and whether the survival functions cross at a certain point in time. In this case, we can use `scurve_tvc` to estimate a Cox model in which the effect of x varies with time and predict the estimated survival function for a specific scenario, here *x=1*, *control1=0* and *control2=0*:

```
. scurve_tvc, gen(S_x) at(x 1 control1 0 control2 0) tvc(x) texp(ln(_t)) replace
(note: variable S_x not found)

Dataset has been temporarily split at failure times
(891 failure times)
(493,614 observations (episodes) created)

The estimation is based on the following Cox Proportional Hazards Model:

         failure _d:  failure
   analysis time _t:  ftime
                 id:  id

Iteration 0:   log likelihood = -5506.5059
Iteration 1:   log likelihood = -5299.4786
Iteration 2:   log likelihood = -5298.9218
Iteration 3:   log likelihood = -5298.9218
Refining estimates:
Iteration 0:   log likelihood = -5298.9218

Cox regression -- no ties

No. of subjects =        1,000                 Number of obs    =     494,614
No. of failures =          891
Time at risk    =  13122.84325
                                               LR chi2(4)       =      415.17
Log likelihood  =   -5298.9218                 Prob > chi2      =      0.0000

------------------------------------------------------------------------------
          _t |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |  -.9929653   .1971679    -5.04   0.000    -1.379407   -.6065234
    control1 |  -1.230347   .0740657   -16.61   0.000    -1.375513   -1.085181
    control2 |  -.3876476   .0352886   -10.99   0.000     -.456812   -.3184833
        _x_t |     .63722   .0861443     7.40   0.000     .4683803    .8060597
------------------------------------------------------------------------------
Note: tvc-interactions denoted by _varname_t were interacted with ln(_t).
```

Note that `scurve_tvc` automatically executes all steps of the workaround presented above. It splits the data at failure times, generates interaction variables based on the function which you specified and shows you the model output. It restores the estimated survival function in a new variable called *S_x*.

We repeat the same process, but this time for the scenario with *x=0*. We then plot this variable against the corresponding analysis time stored in the new variable *_tscurve*. Figure 2 shows that for the specified scenarios, the survival curves cross after about 9 time units. Hence, the negative side-effects of treatment x outweigh its benefits after this time.

```
. quietly scurve_tvc, gen(S_nox) at(x 0 control1 0 control2 0) tvc(x) ///
>         texp(ln(_t)) replace
.
. label var S_x "x=1"
. label var S_nox "x=0"
.
.
. twoway line S_x S_nox _tscurve, c(J J) scheme(sj) title("") ///
>         xtitle("Analysis time") ytitle("Probability of survival") ylab(0(.2)1)
```

If the user is interested in a single scenario only, using the option `graph` as well as `plotopts()` will automatically produces a graph for the scenario specified in `at()`.

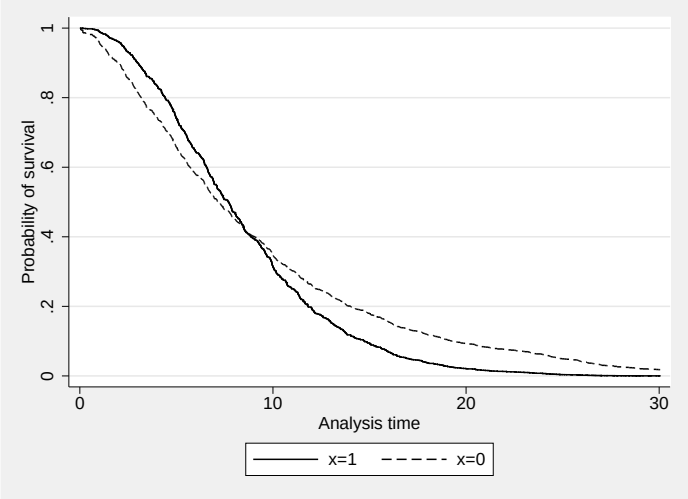

Figure 2: Predicted survival probabilities with and without treatment x. Remaining covariates held at zero.

Often, however, we are interested in a comparison of different scenarios, similar to Putter et al. (2005). Hence, we can use `scurve_tvc` to estimate the survival functions for *x=0* and *x=1* at various values of the remaining covariates. In the hypothetical cancer treatment example, this might imply calculating the estimated survival probabilities for specific patient characteristics with different risk-levels. We can then plot the estimated survival functions for each scenario to transparently communicate the estimated survival probabilities and highlight potential trade-offs for treatments with time-varying effects.

Figure 3 provides these estimates for four scenarios. The substantive interpretation for our hypothetical cancer treatment implies that the treatment x is ineffective and even harmful for low and moderate risk patients. However, high risk patients profit from the procedure. The graphical representation therefore helps to communicate the complex results from the model with non-proportional hazards very clearly and intuitively.

To highlight that the procedure accurately captures the data generating process, Figure 3 shows the analytically calculated, true survival functions and compares these to the estimates from `scurve_tvc`. The plot indicates that the results are in close agreement with underlying data generating process.

4 Conclusion

This paper demonstrates how to estimate survival functions from Cox models with time-varying coefficients. The code is automated in the program `scurve_tvc`. The procedure allows to visualize the predictions of models with non-proportional hazards and enables to effectively communicate model predictions for different covariate scenarios to

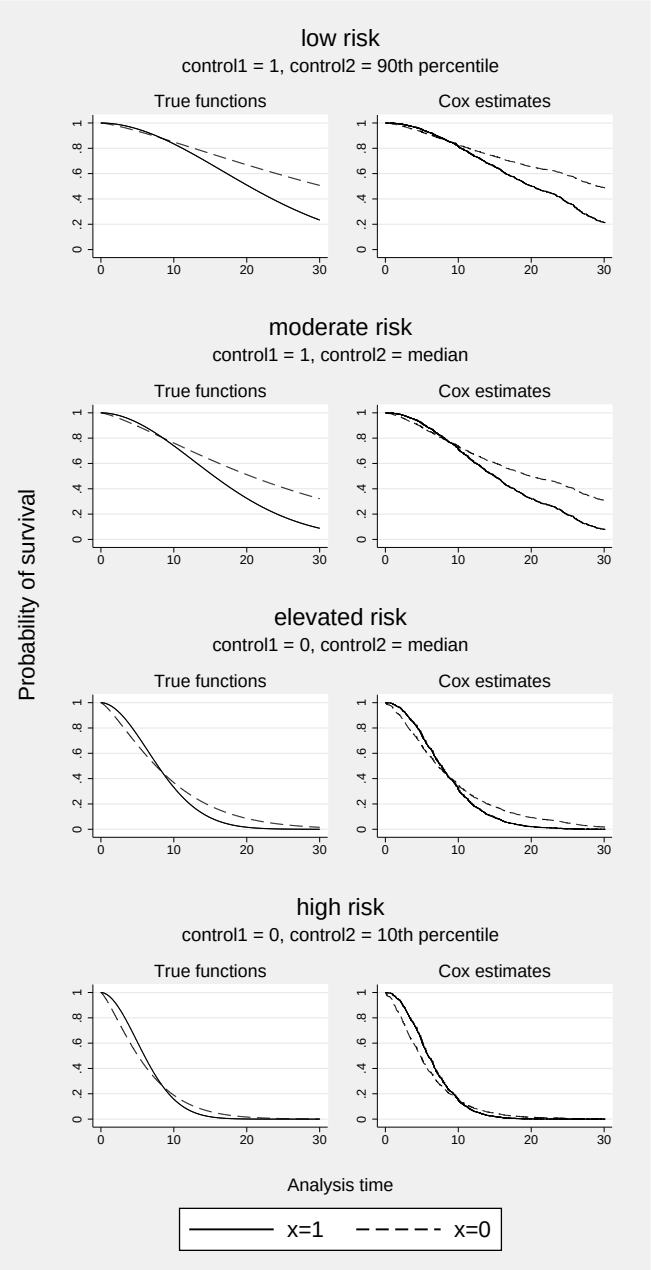

Figure 3: Comparing true, analytically calculated survivor functions (left) for the simulated data with empirical estimates calculated with `scurve_tvc` (right).

a broader audience.

5 References

Box-Steffensmeier, J. M., D. Reiter, and C. Zorn. 2003. Nonproportional Hazards and Event History Analysis in International Relations. *Journal of Conflict Resolution* 47(1): 33–53.

Cleves, M. A., W. Gould, R. G. Gutierrez, and Y. V. Marchenko. 2010. *An Introduction to Survival Analysis Using Stata*. 3rd ed. College Station, Tex.: Stata Press.

Crowther, M. J., and P. C. Lambert. 2012. Simulating complex survival data. *Stata Journal* 12(4): 674–687(14). http://www.stata-journal.com/article.html?article=st0275.

Giolo, S. R., J. E. Krieger, A. J. Mansur, and A. C. Pereira. 2012. Survival analysis of patients with heart failure: Implications of time-varying regression effects in modeling mortality. *PLoS ONE* 7(6): e37392.

Kalbfleisch, J. D., and R. L. Prentice. 2002. *The Statistical Analysis of Failure Time Data*. 2nd ed. Hoboken, N.J.: Wiley-Interscience.

Nilsson, M., and J. Nivre. 2013. Proportional hazards modeling of saccadic response times during reading. *Topics in Cognitive Science* 5(3): 541–563.

Putter, H., M. Sasako, H. H. Hartgrink, van de Velde, C J H, and van Houwelingen, J C. 2005. Long-term survival with non-proportional hazards: results from the Dutch Gastric Cancer Trial. *Statistics in Medicine* 24(18): 2807–2821.

About the authors

Constantin Ruhe is a postdoctoral researcher at the Department of Politics and Administration, University of Konstanz, Germany. His main research interest are the quantitative analysis of political violence and conflict management as well as applied statistics.