```
*===============================================================================
*===============================================================================
* GENERATING SYNTHETIC DATA TO REPLICATE COVARIATE DISTRIBUTIONS AND
* SURVIVAL TIMES FROM A REAL-WORLD DATASET USING MULTINOMAL LOGISTIC REGRESSION,
* INVERSE-RANK-BASED TRANSFORMATIONS AND FLEXIBLE PARAMETRIC MODELLING
*
* CODE AUTHOR: AIDEN SMITH
*
* CORRESPONDING ADDRESS: ajs134@le.ac.uk
*
* LAST UPDATE: 12/10/2021
*
*===============================================================================
*===============================================================================
timer clear 1
timer on 1

 // install required Stata packages
ssc install stpm2, replace
ssc install rcsgen, replace

// Load in real-world dataset
use "https://pclambert.net/data/colon.dta", clear

// Clean data
// (restrict to most recent period of diagnosis, keep only required age range)
keep if year8594 == 1
rename dx diagdate

drop if age > 99
drop if age < 18

// Reformat vital status coding to binary dead/alive coding
drop if status == 4
replace status = 1 if status ==2
label define statuslbl 0 "Alive" 1 "Dead", replace
label values status statuslbl

// stset data to assign as time-to-event in stata
// last exit date (31/12/1995) <- needed for simulated data later
stset exit, id(id) fail(status == 1)  ///
            exit(time mdy(12,31,1995)) ///
            scale(365.25) origin(diagdate)
keep if _st == 1

// generate global macro for last observed exit time for use in
// survival prediction censoring in simulated data
qui su _t
capture drop local maxStime
global maxStime = r(max)
// (last observed exit 10.96 years)

// recode stage unknown to missing and assign missing as unique
// factor variable level to preserve distributions in simulated data
tab stage, missing
replace stage = . if stage == 0
replace stage = 4 if stage ==.

// generate dummy variabe levels for use in surival model
tab stage, gen(stageN)
tab subsite, missing
tab subsite, gen(subsiteN)

// recode sex as 0 and 1
gen female = sex - 1

// rename age
rename age agediag
```

```
69
70  // Winsorize age distribution and keep splines equal for top and bottom
71  // 2% of distribution to aid with model fitting
72  capture drop ageadj
73  _pctile agediag, per(2)
74  global age_cutoff_low `r(r1)'
75  gen ageadj = cond(agediag<$age_cutoff_low , $age_cutoff_low , agediag)
76  _pctile agediag, per(98)
77  global age_cutoff_high `r(r1)'
78  replace ageadj = cond(ageadj>$age_cutoff_high , $age_cutoff_high , ageadj)
79
80  // generate age splines using adjusted age distribution, orthogonalise
81  // and store R matrix
82  capture drop agercs*
83  rcsgen ageadj, df(3) gen(agercs) orthog
84  global ageknots `r(knots)'
85  matrix R = r(R)
86  matrix list R
87
88  // generate interaction terms which will be used in the flexible parametric
89  // survival model
90  forvalues l = 2/4 {
91      gen stage`l'rcs1 = agercs1*stageN`l'
92      gen stage`l'rcs2 = agercs2*stageN`l'
93      gen stage`l'rcs3 = agercs3*stageN`l'
94      gen stage`l'fem = stageN`l'*female
95  }
96
97  forvalues m = 2/4 {
98      gen sub`m'rcs1 = agercs1*subsiteN`m'
99      gen sub`m'rcs2 = agercs2*subsiteN`m'
100     gen sub`m'rcs3 = agercs3*subsiteN`m'
101     gen sub`m'fem = subsiteN`m'*female
102 }
103
104 gen femrcs1 = agercs1*female
105 gen femrcs2 = agercs2*female
106 gen femrcs3 = agercs3*female
107
108 // Fit complex survival model using stpm2 accounting for covariate main
109 // effects, interaction effects and allowing for non-proportional hazards
110 stpm2 stageN2-stageN4 stage?rcs? subsiteN2-subsiteN4       ///
111         stage?fem sub?rcs? agercs* female femrcs?,         ///
112         df(5) scale(h)                                     ///
113         tvc(agercs* stageN2-stageN4)   ///
114         dftvc(3) failconvlininit
115 // store model estimates to restore later and predict survival times
116 estimates store mod_stpm2
117
118 // generate unique ranks for age at diagnosis
119 egen rank = rank(agediag), unique
120 global N = _N
121 // generate inverse normal rankings
122 gen InvRankNormAge = invnormal((rank - 0.5)/${N})
123 // fit splines using inverse normal ranked distribution and store R matrix
124 rcsgen InvRankNormAge, df(5) gen(invrcs) orthog
125 matrix Rmat=r(R)
126 matrix list Rmat
127 local knots=r(knots)
128
129 // regress rankings on initial age variables and store model estimates
130 regress agediag invrcs*
131 estimates store mod_age
132
133 // fit multinomial logistic regression models with sequential increasing in
134 // complexity for factor variables to capture condtional and overall
135 // distributions
136 mlogit yydx c.agediag
```

```
estimate store mod_yydx

mlogit stage i.yydx##c.agediag
estimate store mod_stage

mlogit female i.yydx##i.stage i.yydx##c.agediag
estimates store mod_fem

mlogit subsite i.yydx##i.female i.stage##i.female ///
              i.stage##i.yydx i.yydx##c.agediag  ///
              i.stage##c.agediag i.female##c.agediag
estimate store mod_sub

// generate number of observations to simulate and store as global macro
summ agediag
gen obs = r(N)
global obs = obs

*================================================================================
save "//uol.le.ac.uk/root/staff/home/a/ajs134/My Documents/Stata Simulacrum/Colon Free/Colon_covmod_inv.dta", replace
*================================================================================

// clear dataframe, set simulation seed for reproducability and set number of
// observations (can be same as original data as is being done here with
// global macro, or can be increased/reduced while still preserving proportions
// in covariate patterns/survival time within the data)
clear
set seed 648213
set obs $obs

*================================================================================
// restore age model estimates and predict simulated age covariate
estimates restore mod_age

gen U2 = rnormal()
rcsgen U2, df(5) gen(invrcs) rmatrix(Rmat)
predict agenew
gen agenew2 = (rnormal(agenew, `e(rmse)'))

rename agenew2 agediag

*================================================================================
// restore year of diagnosis model estimates and predict simulated yydx
estimates restore mod_yydx
predict pyydx*
gen uyydx = runiform()

gen yydx = 1*inrange(uyydx, 0, pyydx1) +                                    ///
           2*inrange(uyydx, pyydx1, pyydx1 + pyydx2) +                      ///
           3*inrange(uyydx, pyydx1 + pyydx2, pyydx1 + pyydx2 + pyydx3) +    ///
           4*inrange(uyydx, pyydx1 + pyydx3 + pyydx3,                       ///
                     pyydx1 + pyydx2 + pyydx3 + pyydx4) +                   ///
           5*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4,              ///
                     pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5) +          ///
           6*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5,     ///
                     pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5 + pyydx6) + ///
           7*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5      ///
                     + pyydx6, pyydx1 + pyydx2 + pyydx3 + pyydx4 +          ///
                     pyydx5 + pyydx6 + pyydx7) +                            ///
           8*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5 +    ///
                     pyydx6 + pyydx7, pyydx1 + pyydx2 + pyydx3 + pyydx4 +   ///
                     pyydx5 + pyydx6 + pyydx7 + pyydx8) +                   ///
           9*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5 +    ///
                     pyydx6 + pyydx7 + pyydx8, pyydx1 + pyydx2 + pyydx3 +   ///
                     pyydx4 + pyydx5 + pyydx6 + pyydx7 + pyydx8 + pyydx9) + ///
          10*inrange(uyydx, pyydx1 + pyydx2 + pyydx3 + pyydx4 + pyydx5 +    ///
                     pyydx6 + pyydx7 + pyydx8 + pyydx9, 1)
```

```
204
205 drop uyydx pyydx*
206 drop if yydx == 0
207
208 // recode from 1/10 back to original diagnosis years
209 replace yydx = 1985 if yydx == 1
210 replace yydx = 1986 if yydx == 2
211 replace yydx = 1987 if yydx == 3
212 replace yydx = 1988 if yydx == 4
213 replace yydx = 1989 if yydx == 5
214 replace yydx = 1990 if yydx == 6
215 replace yydx = 1991 if yydx == 7
216 replace yydx = 1992 if yydx == 8
217 replace yydx = 1993 if yydx == 9
218 replace yydx = 1994 if yydx == 10
219
220 tab yydx
221
222 *================================================================================
223 // restore stage model estimates and predict simulated stage at diagnosis
224 estimates restore mod_stage
225 predict pstage*
226 gen ustage = runiform()
227
228 gen stage = 1*inrange(ustage, 0, pstage1) +                    ///
229             2*inrange(ustage, pstage1, pstage1 + pstage2) +    ///
230             3*inrange(ustage, pstage1 + pstage2,               ///
231                       pstage1 + pstage2 + pstage3) +           ///
232             4*inrange(ustage, pstage1 + pstage2 + pstage3, 1)
233
234 drop ustage pstage*
235 drop if stage == 0
236 drop if stage > 4
237 tab stage
238
239 *================================================================================
240 // restore sex model estimates and predict simulated sex covariate
241 estimates restore mod_fem
242 predict pfem
243
244 gen female = runiform()>pfem
245
246 drop pfem
247 tab female
248
249 *================================================================================
250 // restore subsite model estimates and predict simulated subsite covariate
251 estimates restore mod_sub
252 predict psub*
253 gen usub = runiform()
254
255 gen subsite = 1*inrange(usub, 0, psub1) +                  ///
256             2*inrange(usub, psub1, psub1 + psub2) +        ///
257             3*inrange(usub, psub1 + psub2,                 ///
258                       psub1 + psub2 + psub3) +             ///
259             4*inrange(usub, psub1 + psub2 + psub3, 1)
260
261 drop usub psub*
262 drop if subsite == 0
263 drop if subsite > 4
264 tab subsite
265
266
267 *================================================================================
268 // recreate age splines in simulated data to match splines fitted in
269 // original data prior to simulation
270 gen ageadj = cond(agediag<$age_cutoff_low , $age_cutoff_low , agediag)
271 replace ageadj = cond(ageadj>$age_cutoff_high , $age_cutoff_high , ageadj)
```

```
rcsgen ageadj, knots(${ageknots}) rmatrix(R) gen(agercs)

*================================================================================
// generate dummy variables for stage and subsite
tab stage, gen(stageN)
tab subsite, gen(subsiteN)

// recreate interaction terms from original data which were fitted in the
// survival model, these are necessary to restore survival model estimates
// to predict survival times
forvalues l = 2/4 {
    gen stage`l'rcs1 = agercs1*stageN`l'
    gen stage`l'rcs2 = agercs2*stageN`l'
    gen stage`l'rcs3 = agercs3*stageN`l'
    gen stage`l'fem = stageN`l'*female
}

forvalues m = 2/4 {
    gen sub`m'rcs1 = agercs1*subsiteN`m'
    gen sub`m'rcs2 = agercs2*subsiteN`m'
    gen sub`m'rcs3 = agercs3*subsiteN`m'
    gen sub`m'fem = subsiteN`m'*female
}

gen femrcs1 = agercs1*female
gen femrcs2 = agercs2*female
gen femrcs3 = agercs3*female

*================================================================================
// restore survival model estimates
estimates restore mod_stpm2
// generate uniform variable to draw centiles from
gen U = runiform()*100
// generate empty _t variable (needed to draw predictions from the model,
// not important long-term)
gen _t=.
// predict survival time, t, using centiles and chosen centol tolerance
predict t, centile(U) centol(0.0001)

// replace survival times past last known follow-up time as single value
replace t = 11 if t > $maxStime
// assign patients as dead/event experienced if survival time less than
// last known follow-up time
gen dead = 1 if t < 11
// assign patients as alive/event not experienced if survival time was greater
// than last known follow-up time
replace dead= 0 if t== 11
// check vital status distribution and rename
tab dead
rename dead status
// round survival times up to nearest full day (avoid survival times of
// less than 1 day which can affect predictions of other metrics)
replace t = ceil(t*365.25)/365.25
// generate unique id number for each simulated patient
gen id=_n
// drop _t variable as it is no longer needed
drop _t
// generate month and day of diagnosis data using uniform distributions across
// available days in each month
gen mmdx = runiformint(1,12)
gen dddx = runiformint(1,31)
replace dddx = runiformint(1,28) if mmdx == 2
replace dddx = runiformint(1,30) if mmdx == 4 | mmdx == 6 | ///
                                    mmdx == 9 | mmdx == 11
// generate diagnosis date variable using simulated yydx and uniformly
// generated month and day information
gen diagdate = mdy(mmdx,dddx,yydx)
// format for ease of use
```

```
format diagdate %d
// generate survival time in days variable
gen t_days = t * 365.25
// generate exit variable for use in stset
gen exit = diagdate + t_days
// format for ease of use
format exit %d

*================================================================================
// implement censoring if exit date exceeds last known follow-up date
replace status = 0 if exit > mdy(12,31,1995)
// assign last known follow-up date to any patient that has an exit date
// beyond last known follow-up date
replace exit = mdy(12,31,1995) if exit > mdy(12,31,1995)

*================================================================================
// stset data to indicate it is survival data
stset exit, origin(diagdate) id(id) failure(status == 1) ///
        exit(time mdy(12,31,1995)) scale(365.25)

*================================================================================
// clean and format any variables which are going to be preserved in the final
// simulated dataset
tab stage
label variable stage "Stage at Diagnosis"
replace stage =. if stage == 4
replace stage =. if stage == 0
label define stagelbl 1 "Localised" 2 "Regional" 3 "Distant", replace
label values stage stagelbl
tab stage, missing

tab female
rename female sex
label variable sex "Patient Sex"
label define sexlbl 0 "Male" 1 "Female", replace
label values sex sexlbl
tab sex

tab yydx
label variable yydx "Year of Diagnosis"

su agediag
label variable agediag "Age at Diagnosis"
recode agediag (min/44.999=0)(45/59.999=1)(60/74.999=2)(75/max=3), gen(agegrp)
tab agegrp
drop if agegrp > 3

tab status
label variable status "Vital Status"
label define statuslbl 0 "Alive" 1 "Dead", replace
label values status statuslbl

tab subsite
label variable subsite "Anatomical Subsite"
label define subsitelbl 1 "Coecum and Ascending" 2 "Transverse" ///
                        3 "Sigmoid and Descending" 4 "Other and NOS"
label values subsite subsitelbl

label variable dddx "Day of Diagnosis"
label variable mmdx "Month of Diagnosis"
label variable diagdate "Date of Diagnosis"

*================================================================================
// keep only variables required for synthetic dataset
// (data does not have to be stset at the end here but by doing so when opening
// the synthetic dataset survival analysis can be performed immediately without
// having to stset again- hence keeping _st _t0 _d _t etc)
keep subsite stage agediag yydx sex status id exit ///
```

```
408       diagdate dddx mmdx _st _t0 _d _t agegrp
409
410   *===============================================================================
411   // save synthetic dataset
412   save "//uol.le.ac.uk/root/staff/home/a/ajs134/My Documents/Stata Simulacrum/Colon
      Free/Colon_Simmed_INV.dta", replace
413
414   timer off 1
415
416   timer list 1
417
418
419
420
421
422
423
424
425
426
427
428
429
```