2013 Spanish Stata Users Group meeting

Stata logistic regression nomogram generator

WORK IN PROGRESS PAPER

Alexander Zlotnik, Telecom.Eng.
Víctor Abraira Santos, PhD

Ramón y Cajal University Hospital

instituto ramón y cajal de investigación sanitaria irycis

NOTICE

- Nomogram generators for logistic and Cox regression models have been updated since this presentation.
- Download links to the latest program versions (nomolog & nomocox), examples, tutorials and methodological notes are available on this webpage: http://www.zlotnik.net/stata/nomograms/

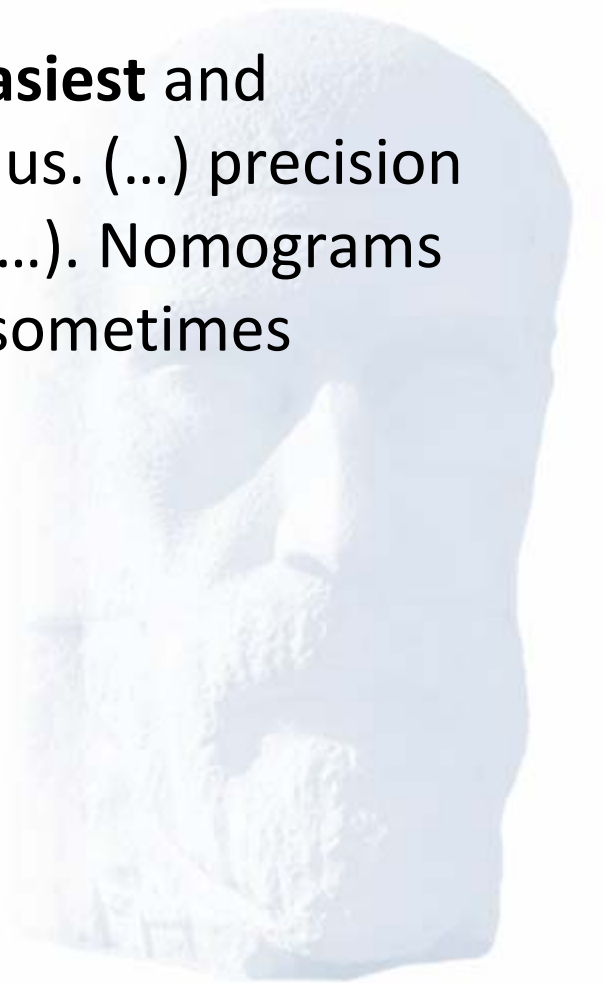
Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Introduction

- Nomograms are one of the **simplest, easiest** and **cheapest** methods of mechanical calculus. (…) precision is similar to that of a logarithmic ruler (…). Nomograms can be used for research purposes (…) sometimes leading to new scientific results.

Source: “Nomography and its applications”
G.S.Jovanovsky, Ed. Nauka, 1977



Introduction

- Examples

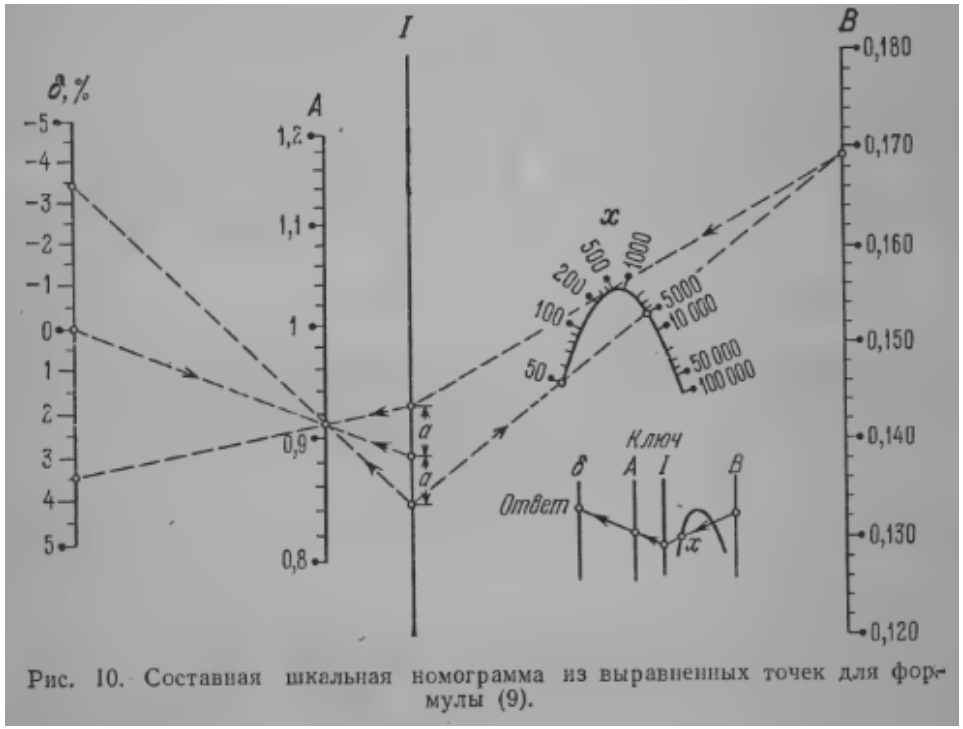
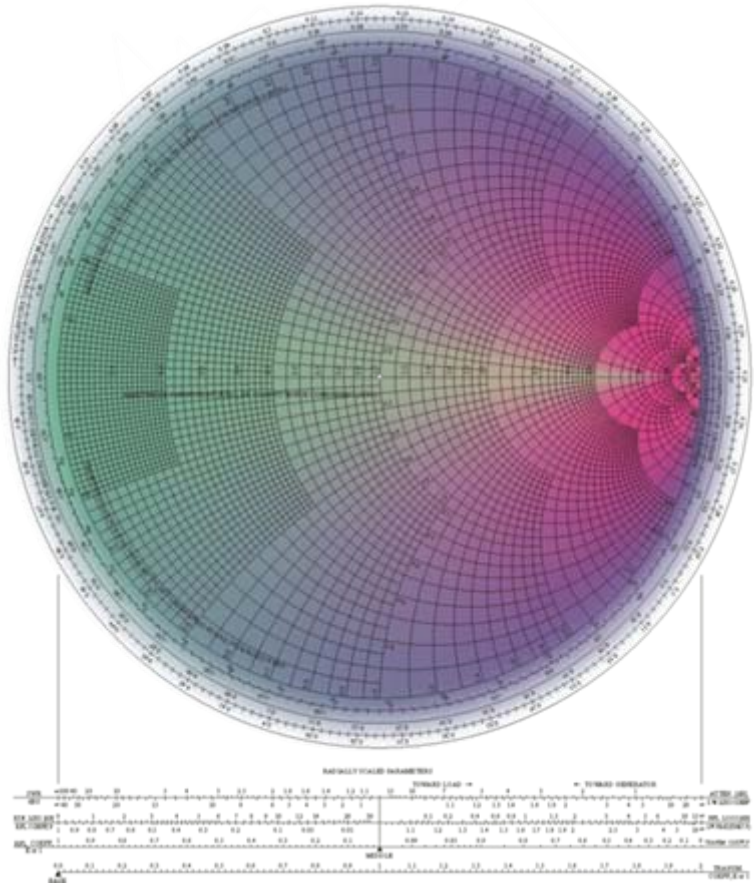


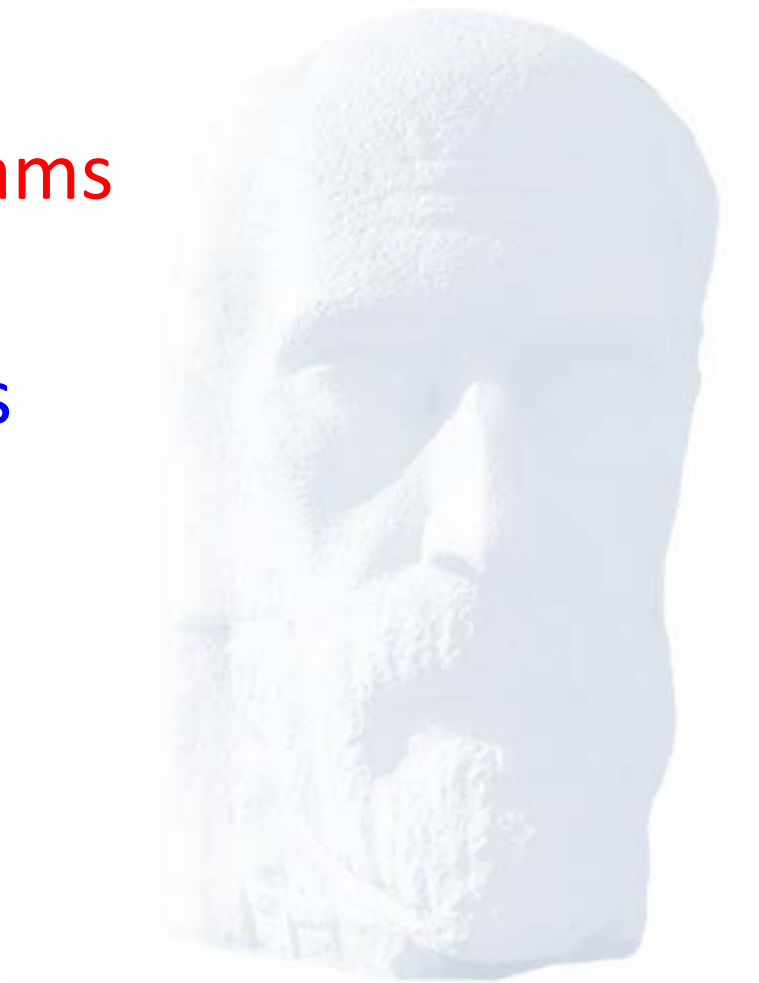
Рис. 10. Составная шкальная номограмма из выравненных точек для формулы (9).

$$\delta = 100\,\frac{\lg x - Ax^B}{\lg x},$$



Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work



Logistic regression predictive models

- Logistic regression-based predictive models are used in many fields, clinical research being one of them.

- Problems:
  - Variable importance is not obvious for some clinicians.
  - Calculating an output probability with a set of input variable values can be laborious for these models, **which hinders their adoption.**

Logistic regression predictive models

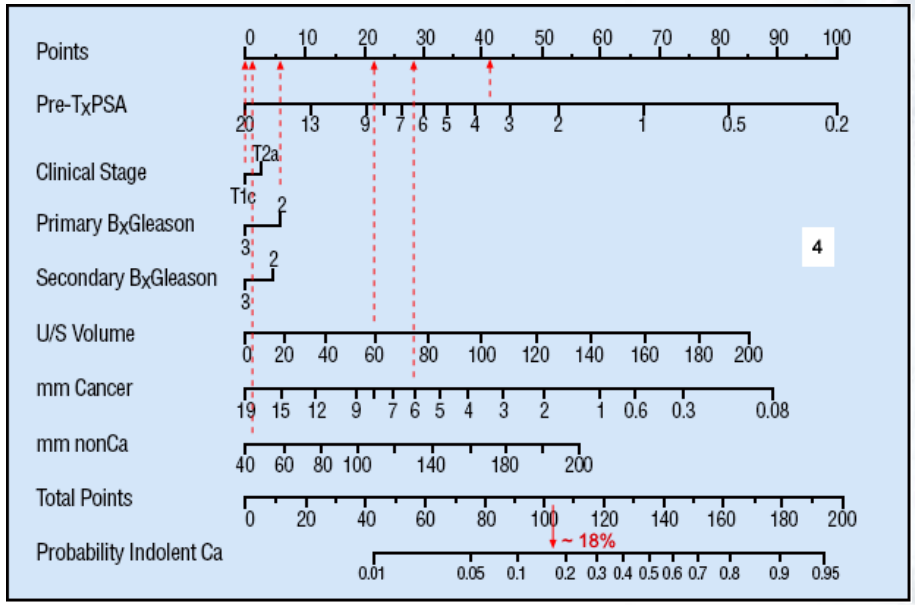
$$\ln\frac{p}{1-p} = Y = -2.903 - 1.698 * \text{Age group[<15]} - 0.453 * \text{Age group[15 - 24]}$$
$$+ 0.233 * \text{Age group[35 - 44]}$$
$$+ 0.423 * \text{Age group[45 - 54]} + 0.580 * \text{Age group[55 - 64]} + 0.921 * \text{Age group[65 - 74]}$$
$$+ 1.280 * \text{Age group[75 - 84]} + 1.662 * \text{Age group[85+]} - 0.127 * \text{Race group[Malay]}$$
$$+ 0.091 * \text{Race group[Indian]} - 0.028 * \text{Race group[Others]} + 0.537 * \text{Arrival mode[ambulance]}$$
$$+ 3.007 * \text{PAC[1]} + 1.488 * \text{PAC[2]} + 0.220 * \text{Prior ED visit in 3 months [Yes]}$$
$$+ 0.360 * \text{Prior hospital admission in 3 months [Yes]} + 0.760 * \text{Chronic conditions [Diabetes only]}$$
$$+ 0.383 * \text{Chronic conditions [Hypertension only]} + 0.633 * \text{Chronic conditions [Dyslipidemia only]}$$
$$+ 0.979 * \text{Chronic conditions [Diabetes with hypertension]} + 0.965 * \text{Chronic conditions [Diabetes with hypertension and dyslipidemia]}$$
$$+ 0.719 * \text{Chronic conditions [Diabetes with dyslipidemia]} + 0.642 * \text{Chronic conditions [Dyslipidemia with hypertension]}$$

$$p = \frac{e^Y}{1 + e^Y}$$

Source: YanSun, 2011

Logistic regression nomograms

- A nomogram could make this calculation much easier. Ex: Kattan nomograms for prostate cancer.

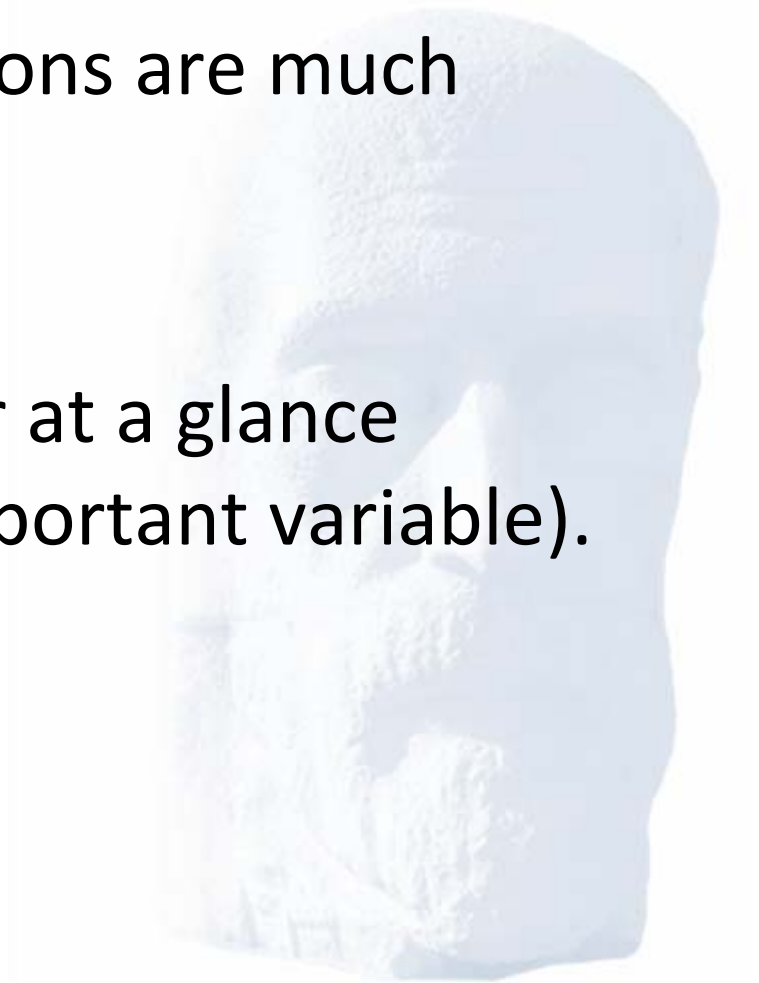


Source: prostate-cancer.org

Logistic regression nomograms

- Output probability calculations are much easier.

- Variable importance is clear at a glance (longer the line => more important variable).



Logistic regression nomograms

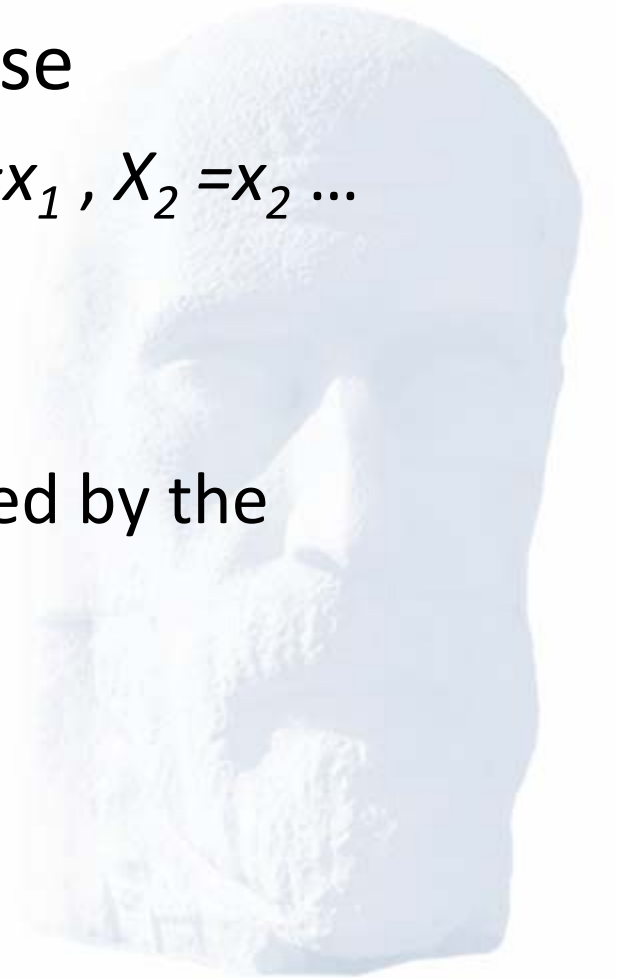
- Logistic regression nomogram generation
  - Plot all possible scores/points ($\alpha_1 x_i$) for each variable ($X_{1..N}$).
  - Get constant (α_0).
  - Transform into **probability of event** given the formula

$$p = \frac{1}{1 + e^{-(\alpha_0 + TP)}}$$

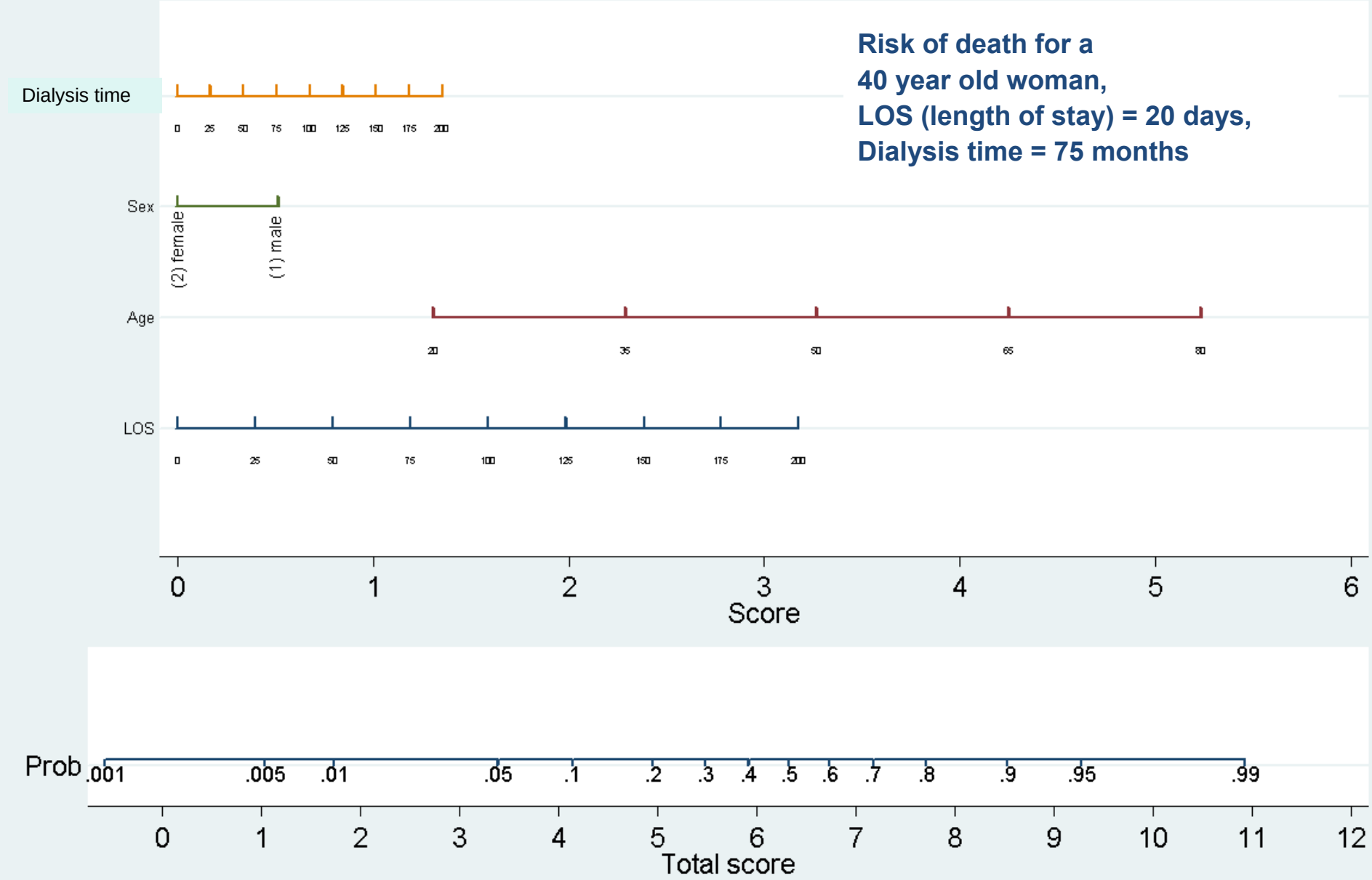
$$Total\ points = TP = \alpha_1 X_1 + \alpha_2 X_2 + \ldots$$

Logistic regression nomograms

- Nomogram usage for a given case
  - Get input variable values, i.e. $X_1 = x_1$, $X_2 = x_2$ …
  - Obtain scores for all variables.
  - Add all scores.
  - Get probability (on a scale adjusted by the constant α_0).

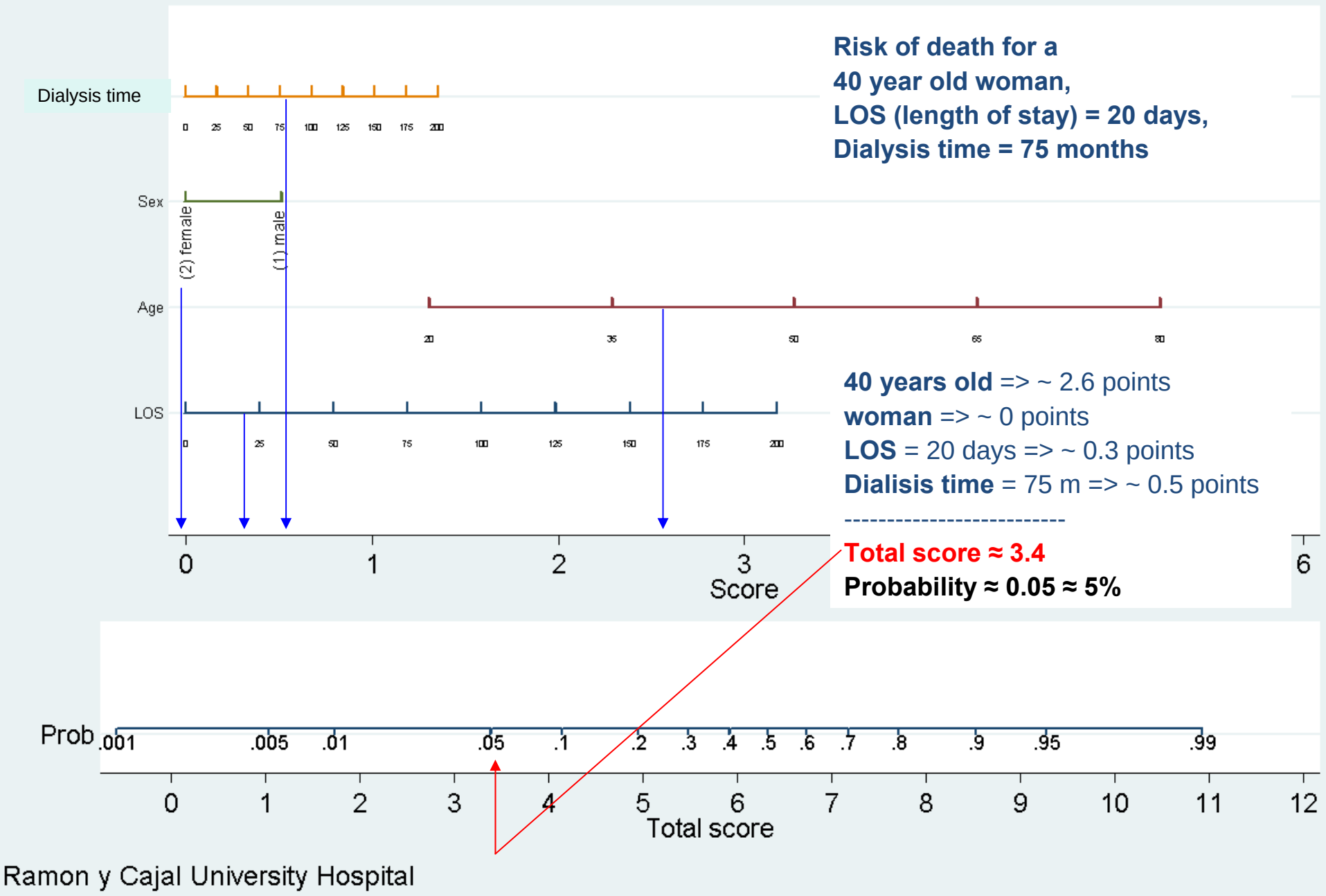


Risk of death for a
40 year old woman,
LOS (length of stay) = 20 days,
Dialysis time = 75 months



Ramon y Cajal University Hospital

instituto ramón y cajal de investigación sanitaria irycis



Risk of death for a
40 year old woman,
LOS (length of stay) = 20 days,
Dialysis time = 75 months

40 years old => ~ 2.6 points
woman => ~ 0 points
LOS = 20 days => ~ 0.3 points
Dialisis time = 75 m => ~ 0.5 points

Total score ≈ 3.4
Probability ≈ 0.05 ≈ 5%

Ramon y Cajal University Hospital

instituto ramón y cajal de investigación sanitaria irycis

Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Objectives

- Build a general-purpose nomogram generator made entirely in Stata without external module dependence. Executable after arbitrary "**logistic**" or "**logit**" Stata commands.

- Automatic (or imposed) variable and data labeling.
- Automatic (or imposed) variable min/max, divisions, variable labels, dummy data labels.

Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Stata programming Gotchas

- **Macro string variables limited to 244 chars.**

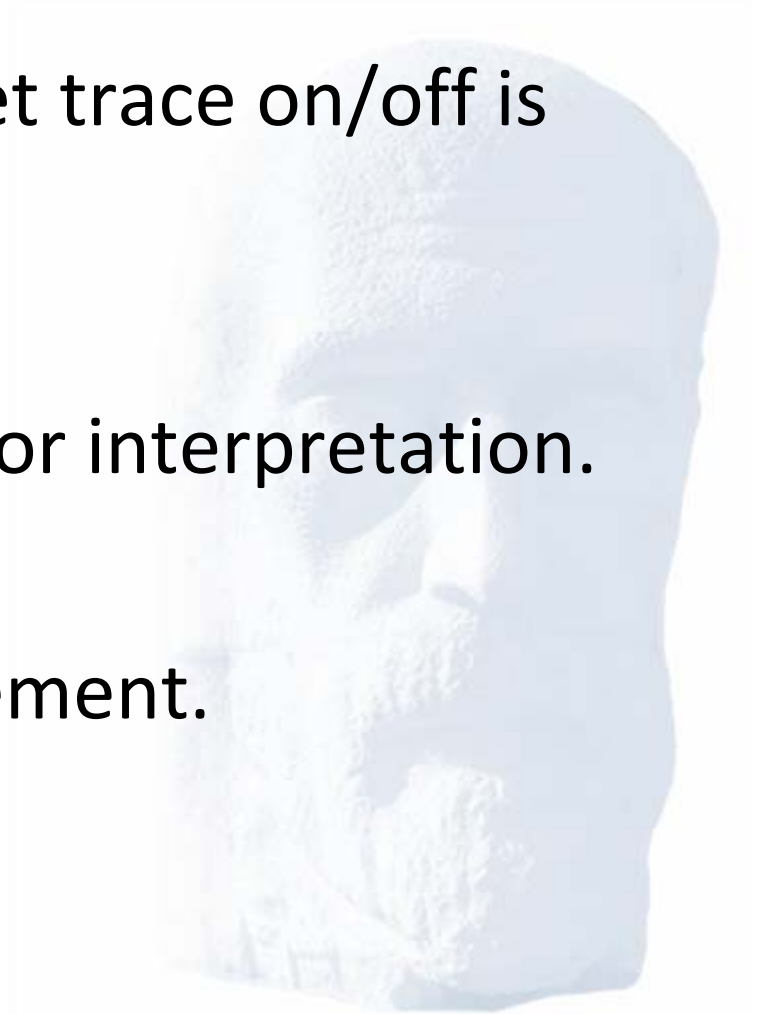
  Solution: strL? Stata 13?

- Hard-to-grasp macro nesting syntax.

- Lack of built-in data structures (non-numeric arrays, dictionaries, lists, etc).

Stata programming Gotchas

- Lack of decent debugger (set trace on/off is not enough).
- Steep learning curve for error interpretation.
- Unit testing is hard to implement.



Structure of Presentation

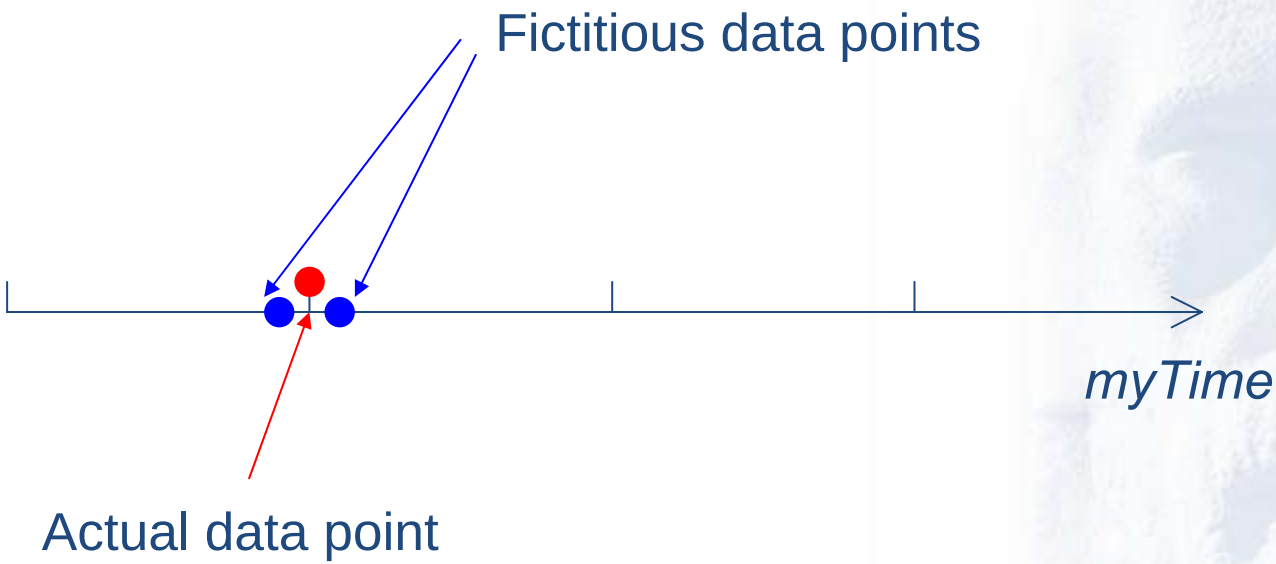
- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Programming techniques

- Time series graph hacks.
- **logit** & **logistic** outputs.
- Variable & data labels.
- Macro nesting.

Programming techniques

- Time series graph hacks




```
xtset Variable myTime
```

Programming techniques

- Logit & logistic outputs

help logistic

Matrices

e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(mns)	vector of means of the independent variables
e(rules)	information about perfect predictors
e(V)	variance-covariance matrix of the estimators
e(V_modelbased)	model-based variance

Scalars

e(N)	number of observations
e(N_cds)	number of completely determined successes
e(N_cdf)	number of completely determined failures
e(k)	number of parameters
e(k_eq)	number of equations in **e(b)**
e(k_eq_model)	number of equations in overall model test
e(k_dv)	number of dependent variables
e(df_m)	model degrees of freedom
e(r2_p)	pseudo-R-squared

instituto ramón y cajal de investigación sanitaria IFYCIS

Programming techniques

- Custom data structures

```
//========= Print custom data structure START =========
if `iDebug' > 0 {
 forvalue i=1/`iNVars' {
  display "--+--"
  display "local asVars_`i'_varkey=" `asVars_`i'_varkey'
  display "local asVars_`i'_varname=" "`asVars_`i'_varname'"
  display "local asVars_`i'_varname_raw=" "`asVars_`i'_varname_raw'" //debug purposes
  display "local asVars_`i'_varlabel=" "`asVars_`i'_varlabel'"
  display "local asVars_`i'_varlabeldisp=" "`asVars_`i'_varlabeldisp'"
  display "local asVars_`i'_type=" `asVars_`i'_type'
  display "local asVars_`i'_min=" `asVars_`i'_min'
  display "local asVars_`i'_max=" `asVars_`i'_max'
  display "local asVars_`i'_divs=" `asVars_`i'_divs'
  display "local asVars_`i'_ncoefs=" `asVars_`i'_ncoefs'
  display "local asVars_`i'_refcoef=" `asVars_`i'_refcoef'

  if ! missing("`asVars_`i'_ncoefs'") {
    forvalue j=1/`asVars_`i'_ncoefs' {
     if(! missing("`asVars_`i'_coef_`j'_value'")) {
      if `asVars_`i'_coef_`j'_value' != . {
       display "local asVars_`i'_coef_`j'_value=" `asVars_`i'_coef_`j'_value'
       display "local asVars_`i'_coef_`j'_label=" "`asVars_`i'_coef_`j'_label'"
       display "local asVars_`i'_coef_`j'_labeldisp=" "`asVars_`i'_coef_`j'_labeldisp'"
     } //END if `asVars_`i'_coef_`j'_value' != .
    } //END if(! missing("`asVars_`i'_coef_`j'_value'"))
   } //END forvalue j=1/`asVars_`i'_ncoefs'
  } //END if ! missing("`asVars_`i'_type'")
 } //END forvalue i=1/`iNVars'
} //END if `iDebug'
//========= Print custom data structure END =========
```

Programming techniques

- Variable & data labels

```
matrix rcoefs = e(b)
local temp: colnames rcoefs
local rvar_names = substr("`temp'", 1, length("`temp'") - 6)
```

```
if "`sLastDummy'" != "`sThisDummy'" {
  levelsof `sThisDummy', local(`sThisDummy'_levels)
  local qq = 0
  foreach val of local `sThisDummy'_levels {
   local svTempDummy`j'_DataLabel`q' : label `sThisDummy' `val'
   local qq = `qq' + 1
  } // end foreach
```


instituto ramón y cajal de investigación sanitaria irycis

Programming techniques

- Macro nesting

```
forvalues i=1/`asVars_`j'_ncoefs' {
 if(! missing("`asVars_`j'_coef_`i'_value'")) {
 if `asVars_`j'_coef_`i'_value' != 0 & `asVars_`j'_coef_`i'_value' != . {

  local `j'x`i'=`asVars_`j'_coef_`i'_value' * 1000

  if ``j'x`i'' >= 0 {
   local sData  `" ``j'x`i''  "'

   local sValueLabel = "`asVars_`j'_coef_`i'_labeldisp'"
   local sTemp  `" text (`iYPos' `sData' "`sValueLabel'", size(1.6) orient(vertical))"'
  }
  else {
   local `j'x`i' = (-1) * (``j'x`i'')
   local sData  `" ``j'x`i''  "'

   local sValueLabel = "-`asVars_`j'_coef_`i'_labeldisp'"
   local sTemp  `" text (`iYPos' `sData' "`sValueLabel'", size(1.6) orient(vertical))"'
  }
 } //END if `asVars_`j'_coef_`i'_value' != 0 & `asVars_`j'_coef_`i'_value'
 } //END if(! missing("`asVars_`j'_coef_`k'_value'"))
 local sNVDM_`j'_`i' = `"`sNVDM_``j''_``i'''  `sTemp'"'
} //END forvalues i=1/`nd'
} //END if `asVars_`j'_type' == 2 {
```

Structure of Presentation

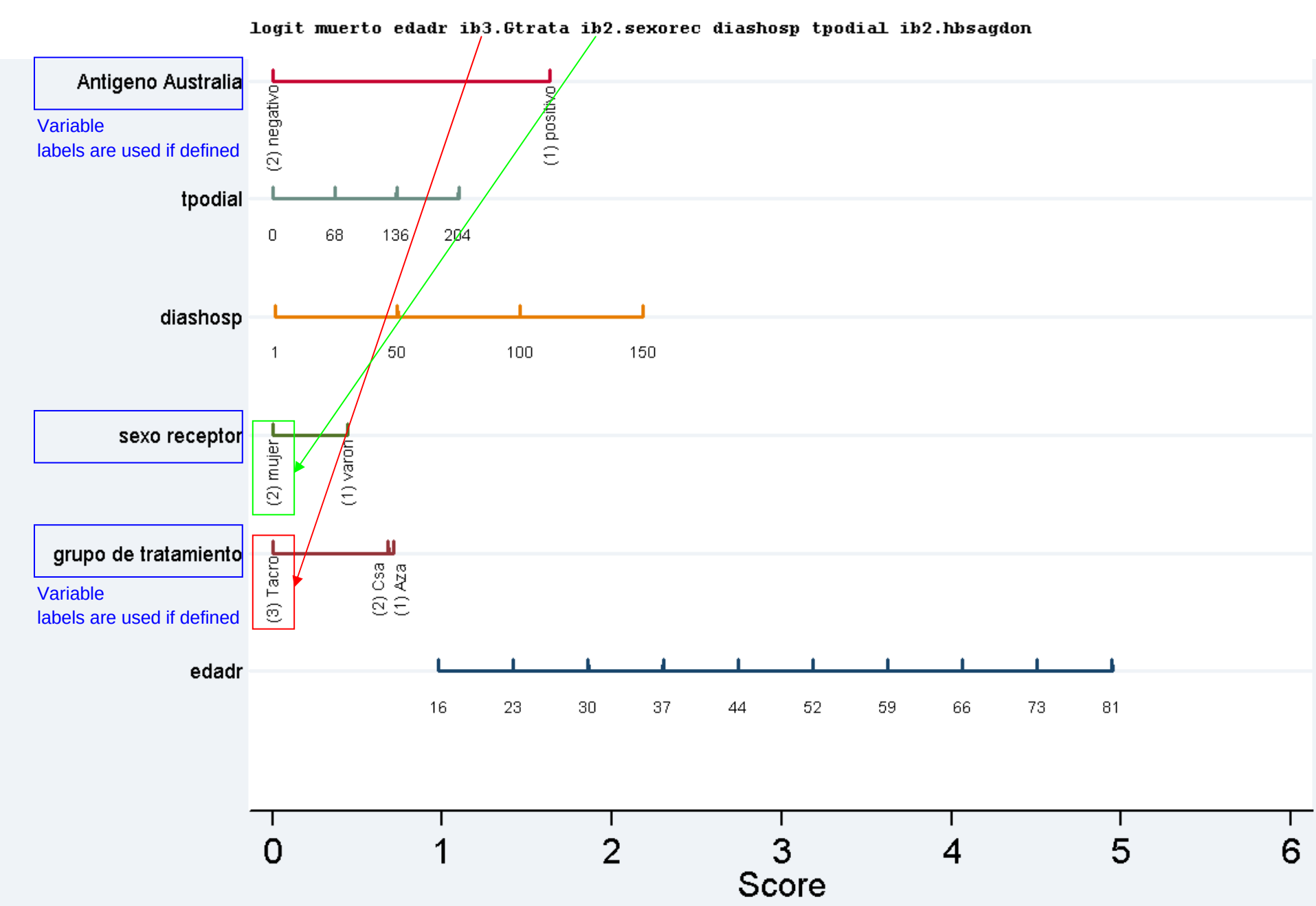
- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Execution example

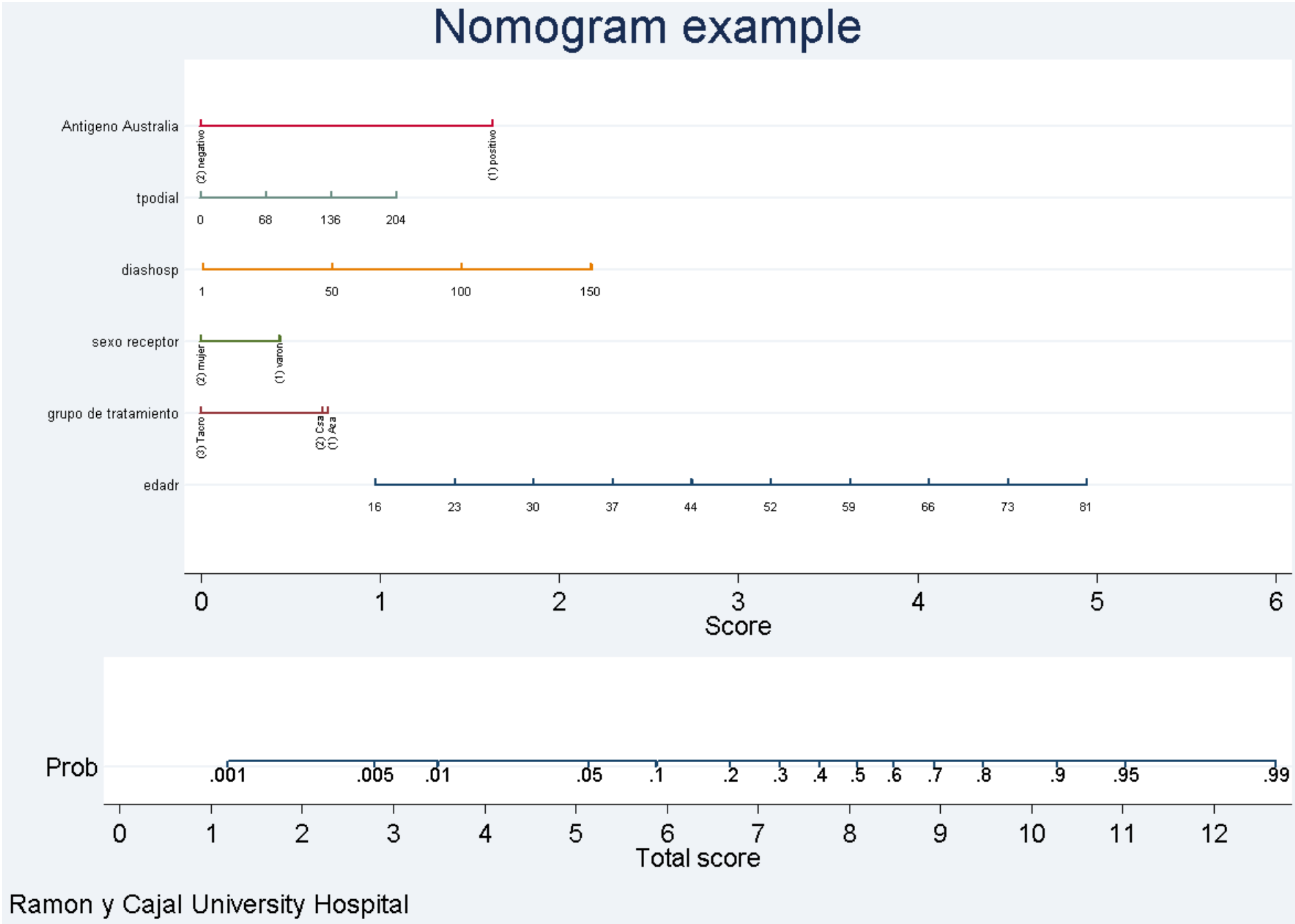
```
. logit muerto edadr ib3.Gtrata ib2.sexorec diashosp tpodial ib2.hbsagdon

Iteration 0:   log likelihood = -564.05484
Iteration 1:   log likelihood = -491.72691
Iteration 2:   log likelihood = -480.05199
Iteration 3:   log likelihood = -479.70677
Iteration 4:   log likelihood = -479.70543
Iteration 5:   log likelihood = -479.70543

Logistic regression                             Number of obs     =       1305
                                                LR chi2(7)        =     168.70
                                                Prob > chi2       =     0.0000
Log likelihood = -479.70543                     Pseudo R2         =     0.1495

------------------------------------------------------------------------------
     muerto |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
------------+-----------------------------------------------------------------
      edadr |   .0611037   .0073464     8.32   0.000     .0467051    .0755023
            |
     Gtrata |
          1 |   .7109415   .4916861     1.45   0.148    -.2527455    1.674628
          2 |   .6817164   .4203042     1.62   0.105    -.1420647    1.505497
            |
  1.sexorec |   .4424468   .1863825     2.37   0.018     .0771437    .8077498
   diashosp |   .0145514   .0040296     3.61   0.000     .0066536    .0224492
    tpodial |   .0053807   .0025295     2.13   0.033      .000423    .0103384
1.hbsagdon |   1.631191   .4346622     3.75   0.000      .779269    2.483114
      _cons |  -8.085328   .6954642   -11.63   0.000    -9.448413   -6.722243
------------------------------------------------------------------------------
```

```
. run "nomo_generator.do"
```


instituto ramón y cajal de investigación sanitaria irycis


logit muerto edadr ib3.Gtrata ib2.sexorec diashosp tpodial ib2.hbsagdon
Antigeno Australia
Variable
labels are used if defined
(2) negativo
(1) positivo
tpodial
0
68
136
204
diashosp
1
50
100
150
sexo receptor
(2) mujer
(1) varon
grupo de tratamiento
(3) Tacro
(2) Csa
(1) Aza
Variable
labels are used if defined
edadr
16
23
30
37
44
52
59
66
73
81
0
1
2
3
4
5
6
Score

```
. run "nomo_generator.do"
```



Ramon y Cajal University Hospital

Positive coefficients

- Usually, positive coefficients are required in these nomograms in order to ease calculations (no substractions to get total score).
- Due to the linear nature of the TP term, it is easy to make all coefficients positive.

$$p = \frac{1}{1 + e^{-(\alpha_0 + TP)}}$$

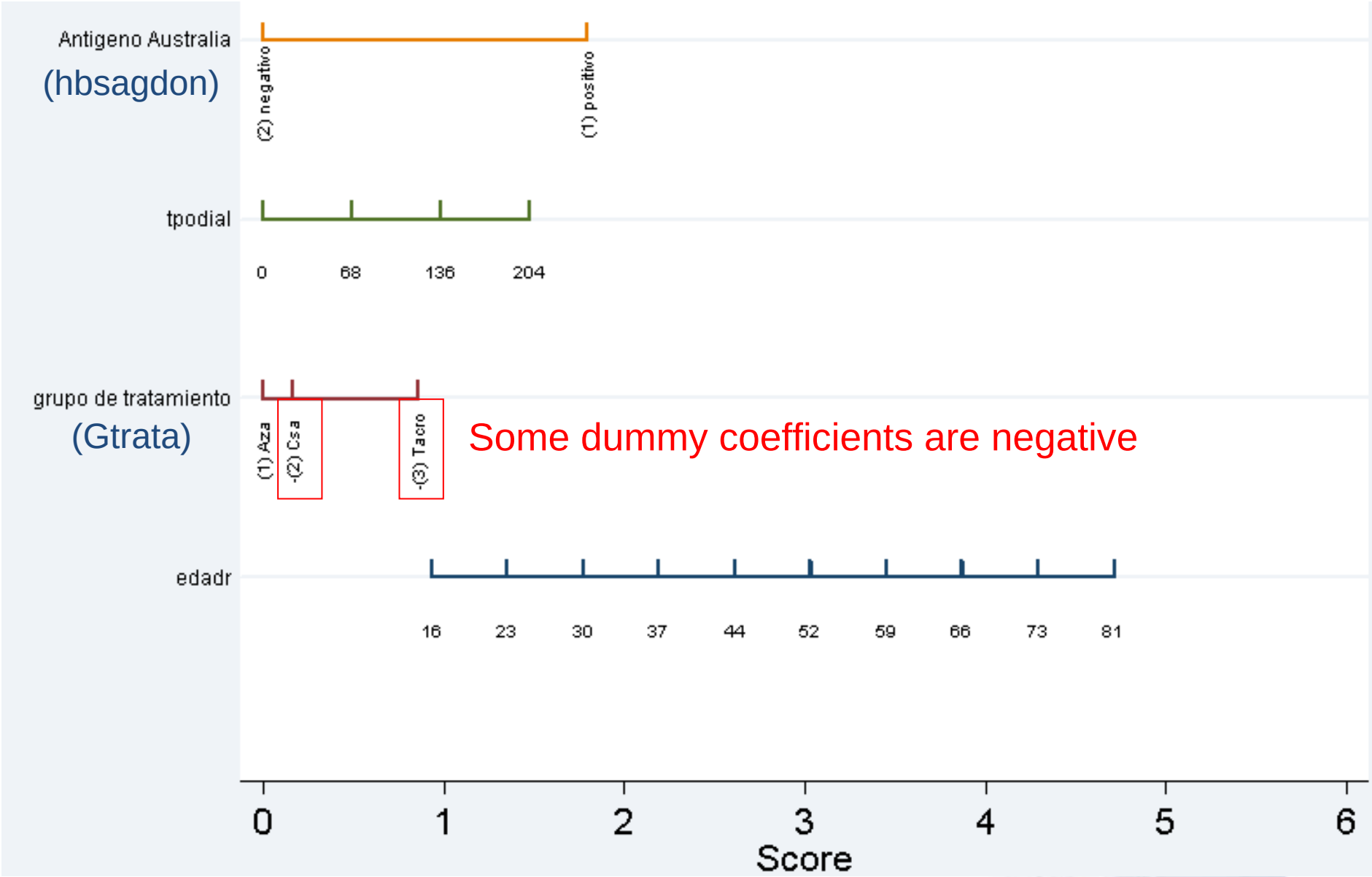
$$TP = \alpha_1 X_1 + \alpha_2 X_2 + \ldots$$

Positive coefficients

- This can be done manually.

- Let's see an example…

  logit muerto edadr ib1.Gtrata tpodial ib2.hbsagdon


Antigeno Australia
(hbsagdon)
(2) negativo
(1) positivo
tpodial
0
68
136
204
grupo de tratamiento
(Gtrata)
(1) Aza
-(2) Csa
-(3) Tacro
Some dummy coefficients are negative
edadr
16
23
30
37
44
52
59
66
73
81
0
1
2
3
4
5
6
Score

Positive coefficients

- The most negative coefficient in Gtrata is “-(3) Tacro”, i.e. data value 3.
- Hence, instead of

  logit muerto edadr ib1.Gtrata tpodial ib2.hbsagdon

  we use…

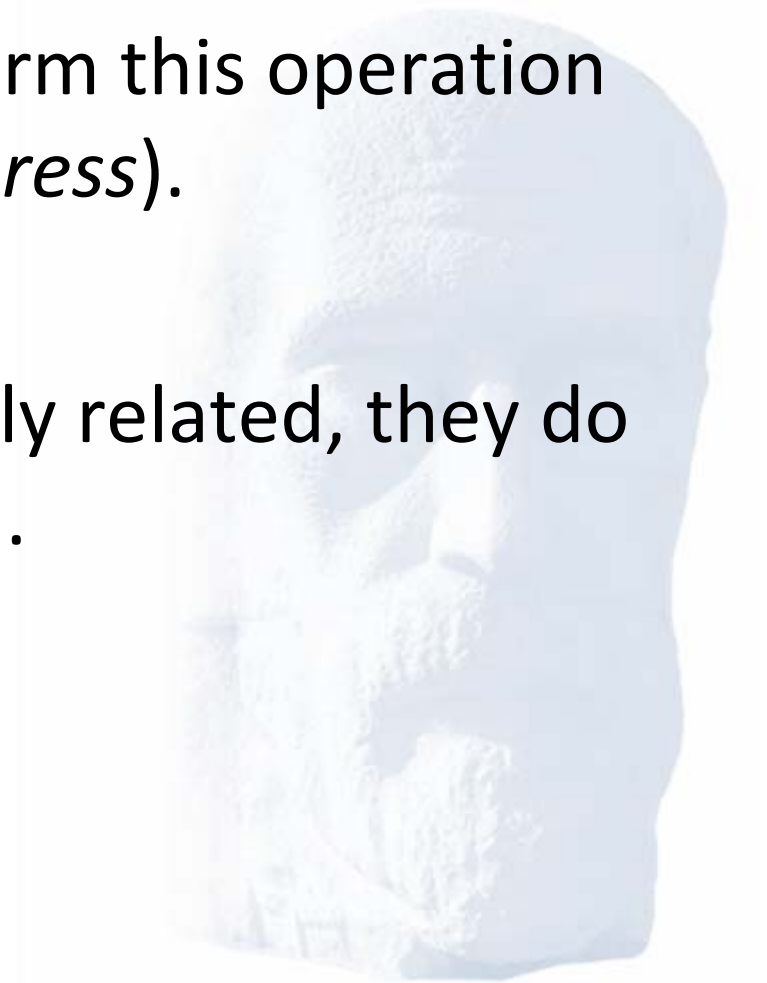
  logit muerto edadr ib3.Gtrata tpodial ib2.hbsagdon

Now all dummy coefficients are positive

Positive coefficients

- The program also can perform this operation automatically (*work in progress*).

- Since coefficients are linearly related, they do not need to be recalculated.



Results

- Supports any kind of variable ordering.
- Supports negative coefficients.
- Supports omitted variables due to collinearity.

- **Works after an almost arbitrary regression command.**
- Let’s see execution modes & parameters…

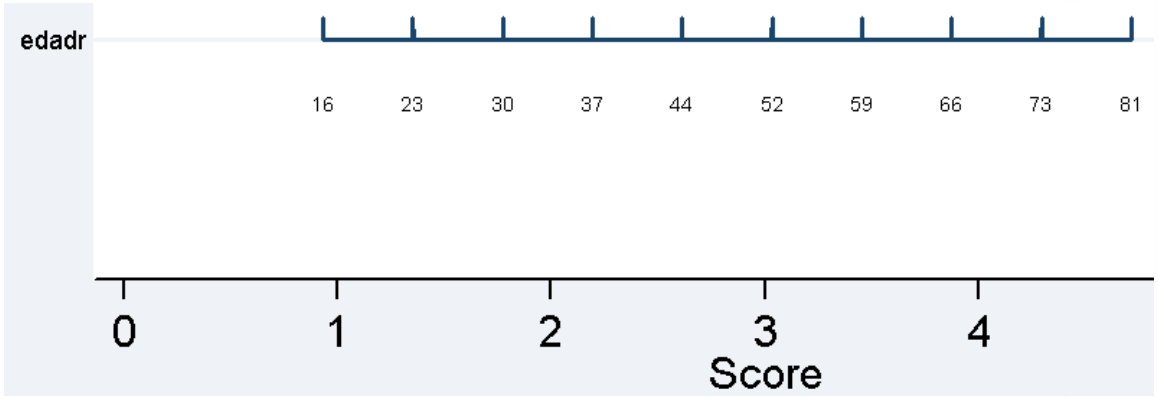
Remember the custom data structure

```
//========= Print custom data structure START =========
if `iDebug' > 0 {
 forvalue i=1/`iNVars' {
  display "--+--"
  display "local asVars_`i'_varkey=" `asVars_`i'_varkey'
  display "local asVars_`i'_varname=" "`asVars_`i'_varname'"
  display "local asVars_`i'_varname_raw=" "`asVars_`i'_varname_raw'" //debug purposes
  display "local asVars_`i'_varlabel=" "`asVars_`i'_varlabel'"
  display "local asVars_`i'_varlabeldisp=" "`asVars_`i'_varlabeldisp'"
  display "local asVars_`i'_type=" `asVars_`i'_type'
  display "local asVars_`i'_min=" `asVars_`i'_min'
  display "local asVars_`i'_max=" `asVars_`i'_max'
  display "local asVars_`i'_divs=" `asVars_`i'_divs'
  display "local asVars_`i'_ncoefs=" `asVars_`i'_ncoefs'
  display "local asVars_`i'_refcoef=" `asVars_`i'_refcoef'

  if ! missing("`asVars_`i'_ncoefs'") {
    forvalue j=1/`asVars_`i'_ncoefs' {
     if(! missing("`asVars_`i'_coef_`j'_value'")) {
      if `asVars_`i'_coef_`j'_value' != . {
       display "local asVars_`i'_coef_`j'_value=" `asVars_`i'_coef_`j'_value'
       display "local asVars_`i'_coef_`j'_label=" "`asVars_`i'_coef_`j'_label'"
       display "local asVars_`i'_coef_`j'_labeldisp=" "`asVars_`i'_coef_`j'_labeldisp'"
      } //END if `asVars_`i'_coef_`j'_value' != .
    } //END if(! missing("`asVars_`i'_coef_`j'_value'"))
   } //END forvalue j=1/`asVars_`i'_ncoefs'
  } //END if ! missing("`asVars_`i'_type'")
 } //END forvalue i=1/`iNVars'
} //END if `iDebug'
//========= Print custom data structure END =========
```

Execution modes

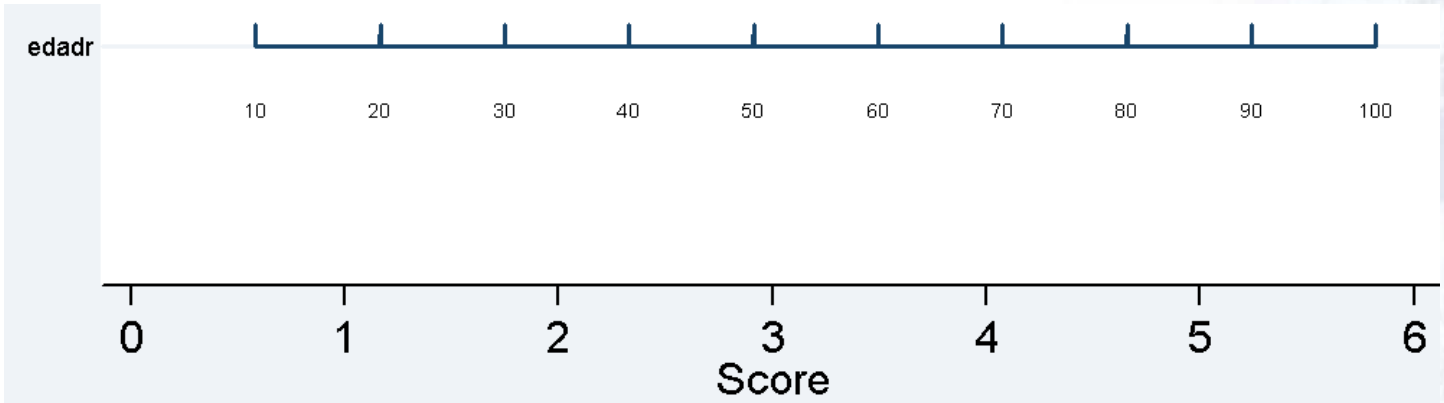
- **Automatic:** everything is determined automatically.
- **Manual:** define all parameters manually (laborious).
- **Hybrid:** get some stuff automatically and refine it manually. For example: variable range is 14 to 81 and we want to make it 10 to 100 leaving all other variables as they are.

Hybrid mode




```
local asVars_1_min = 10
local asVars_1_max = 100
local asVars_1_divs = 10
```

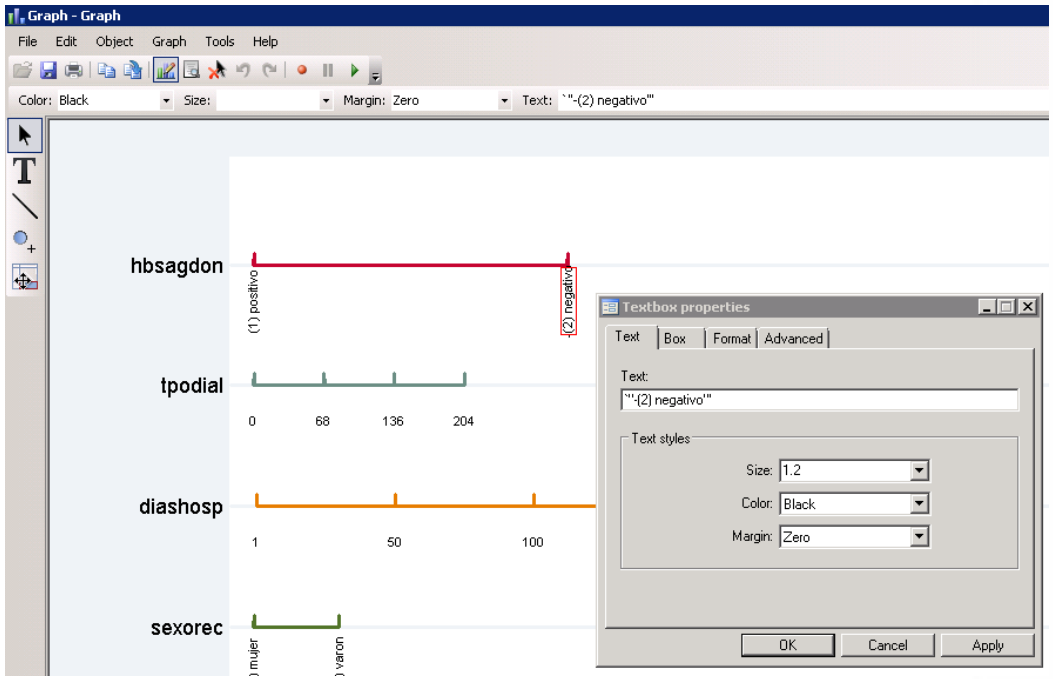
The “edadr” coefficient does not change



instituto ramón y cajal de investigación sanitaria irycis

By the way…

- Since the output is a standard Stata graph, it may be edited for further adjustments.



Other execution options

- **iMaxVarLabelLen = 30** //Max N of chars to display in variable labels
- **iMaxDataLabelLen = 30** //Max N of chars to display in data value labels
- **iVarLabDescr = 0**
  //Use variable description as variable label when possible (0=no; 1=yes)
- **iDummyLabWithValues = 1** //Show data values on dummy data value labels (0=no; 1=yes)
- **iCoefForcePositive = 0** //Force positive coefs (0=no; 1=yes)

Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Limitations

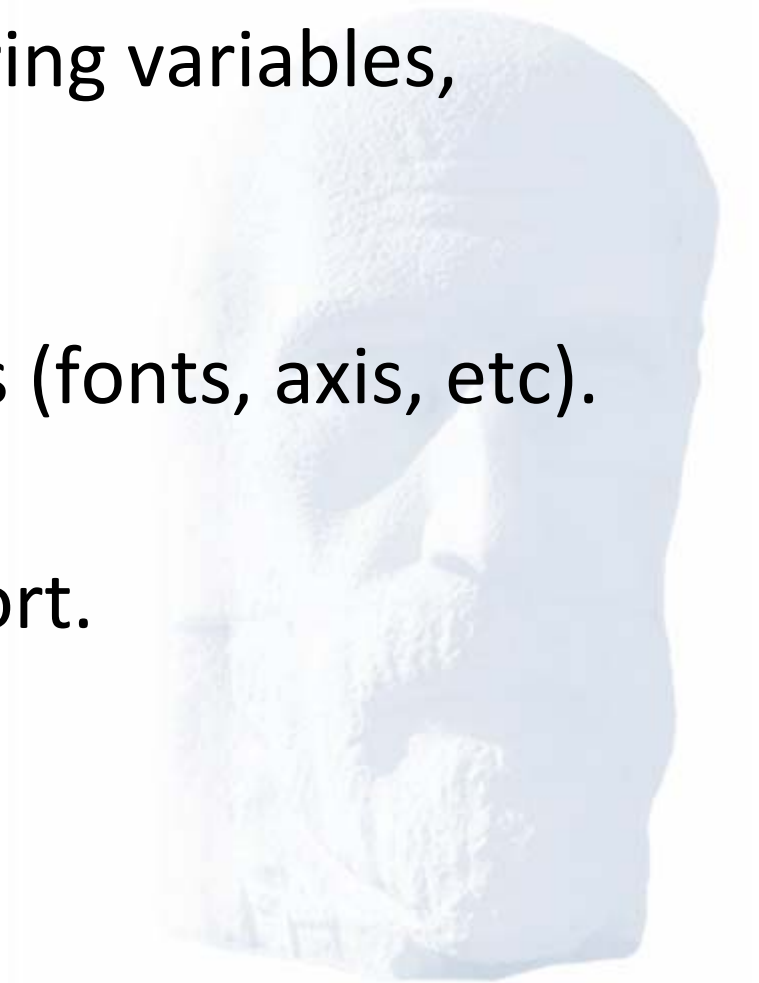
- Dummy syntax must be “bx.var”.
- No interaction operators (“#”, “##”) allowed.
- Maximum of 15 variables (xtline command *options overflow*). Suggestions?
- Categorical variables have a limit of with 40 dummys each (easy to supersede if needed).
- Several performance improvements possible (max. string length).

Structure of Presentation

- Introduction
- Logistic regression nomograms
- Objectives
- Stata programming Gotchas
- Programming techniques
- Results
- Limitations
- Future work

Future work

- Overcoming Stata limits: string variables, xtline command.
- Better drawing adjustments (fonts, axis, etc).
- Interaction operators support.
- Cox regression nomograms.



Questions?

Backup slides

Dummy coefficient re-adjustment

- *Given a categorical variable "A" with N cátegories and a regression constant* α_0

  $z = \alpha_0 + \alpha_{A1} \cdot D_1 + \alpha_{A2} \cdot D_2 + \ldots + \alpha_{AN} \cdot D_N$

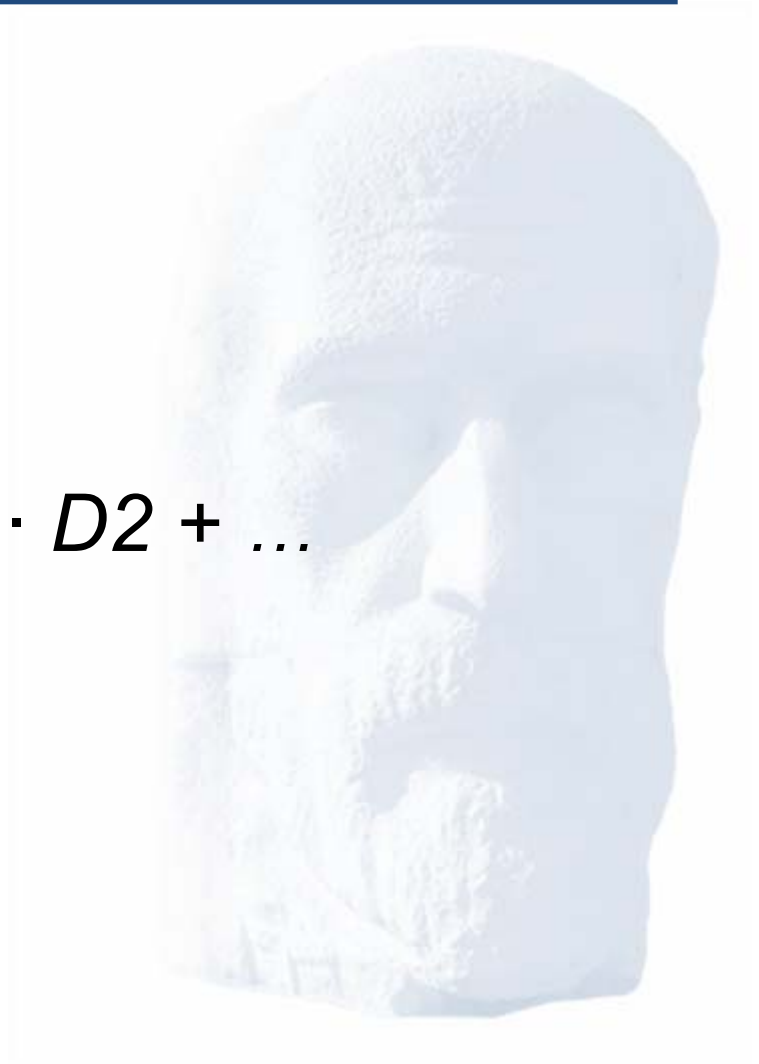
  If $\alpha_{Ai\ i=1..N} < 0$,

  we set as reference the most negative coefficient i.e. $\min(\alpha_{Ai\ i=1..N})$

Dummy coefficient re-adjustment

$$p = \frac{1}{1 + e^{-(\alpha_0 + TP)}}$$

$$TP = \alpha_{A1} \cdot D1 + \alpha_{A2} \cdot D2 + \ldots$$



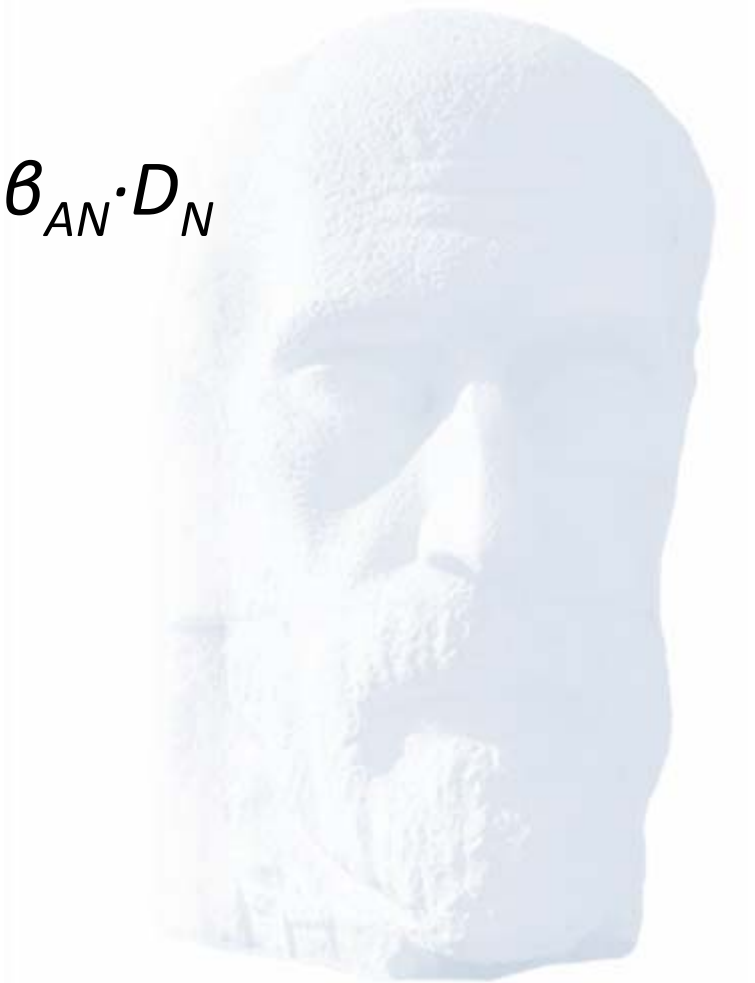
Dummy coefficient re-adjustment

- *And then*

$$z = \beta_0 + \beta_{A1} \cdot D_1 + \beta_{A2} \cdot D_2 + \ldots + \beta_{AN} \cdot D_N$$

where

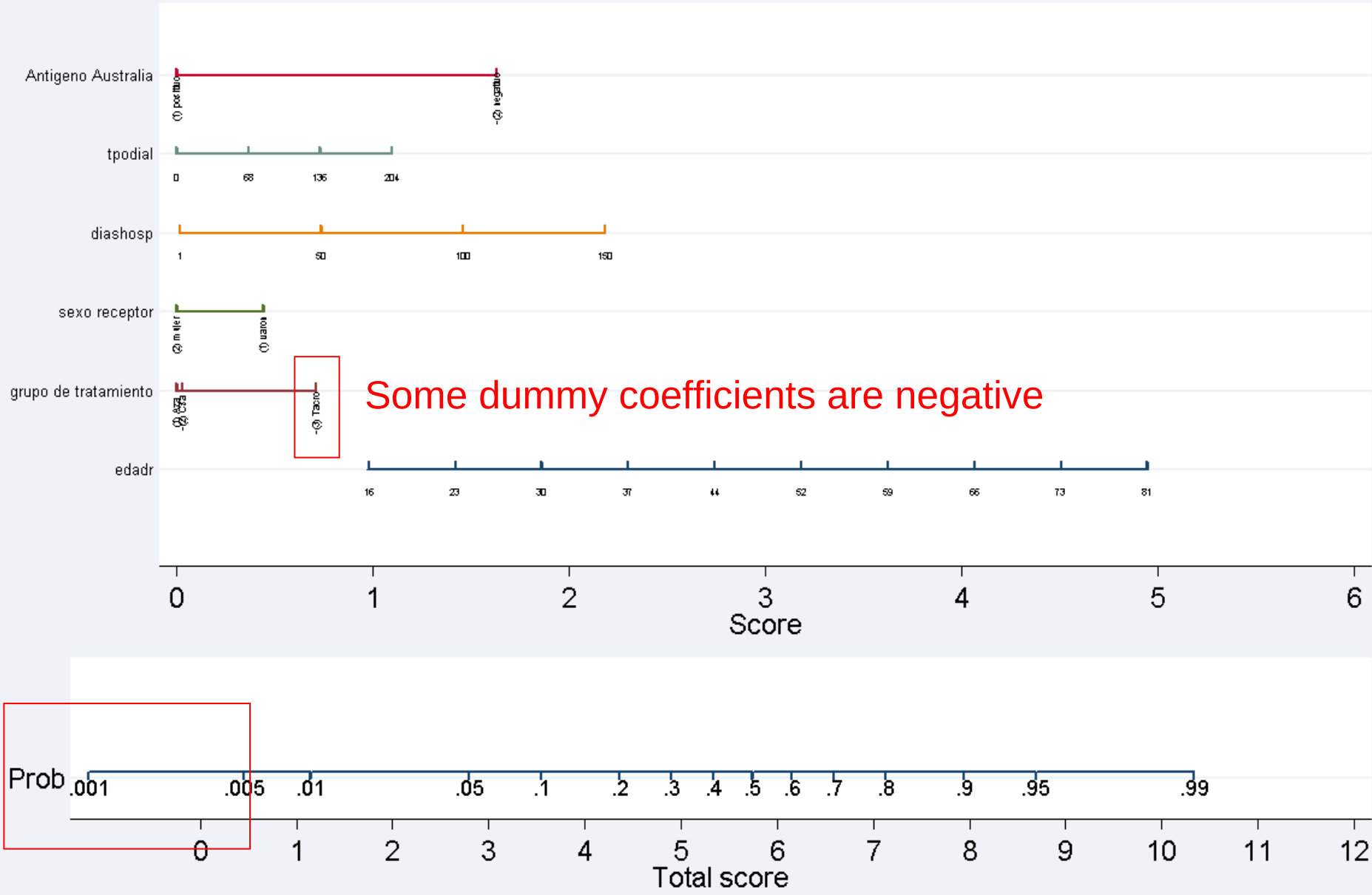
$$\beta_0 = \alpha_0 - min(\alpha_{Ai\ i=1..N})$$
$$\beta_1 = \alpha_1 - min(\alpha_{Ai\ i=1..N})$$
$$\ldots$$
$$\beta_N = \alpha_N - min(\alpha_{Ai\ i=1..N})$$



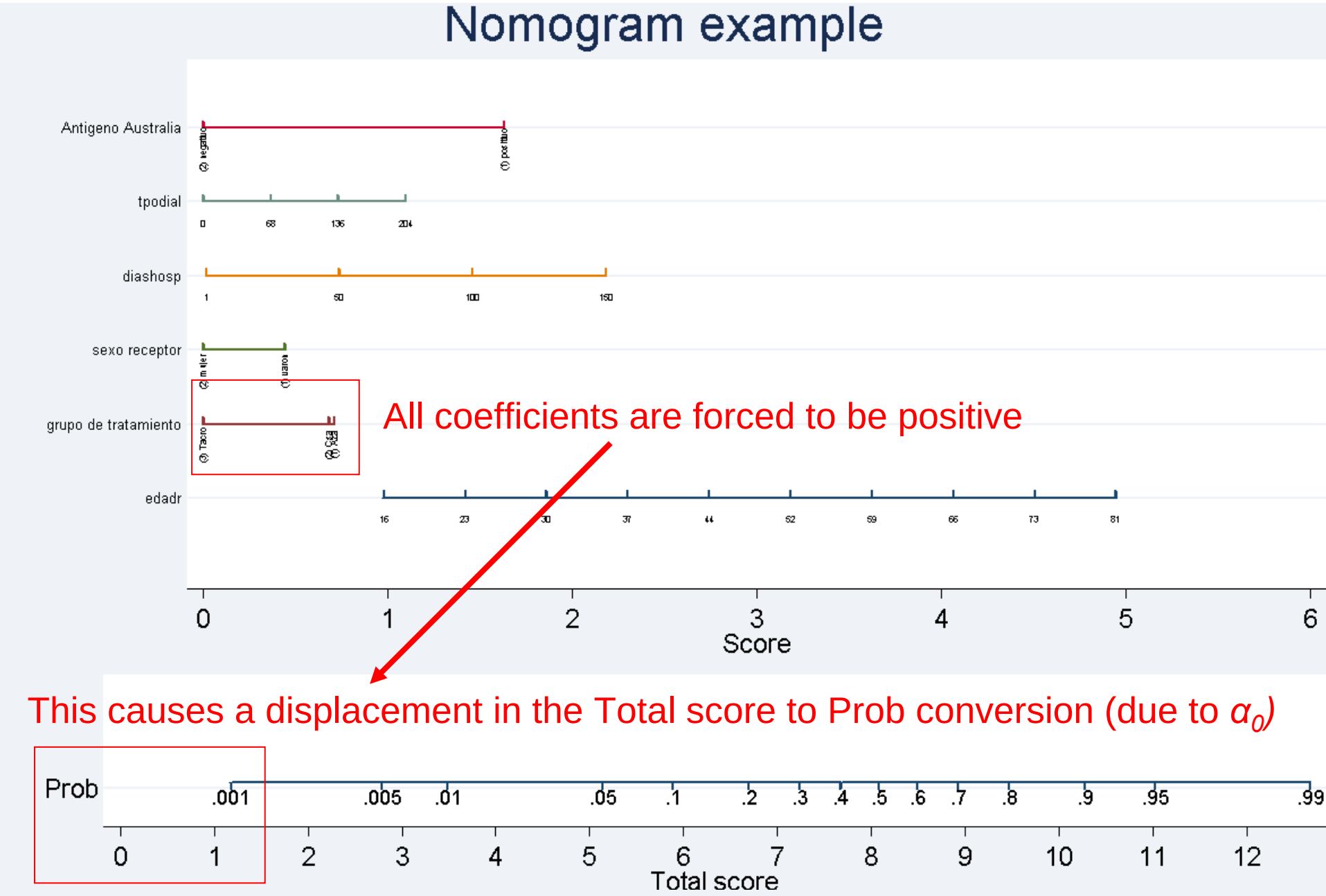
instituto ramón y cajal de investigación sanitaria irycis

Normal execution

Nomogram example



Execution with forced positive coefficients



Ramon y Cajal University Hospital