## Outline of Workflow

**10/14/14:**

Needs:

* Easy to change a parameter and run a new model
* Easy to loop through a series of different parameters and run a new model
* Easy to quantify simulation uncertainty: is this necessary?

New workflow idea:

* Use a wrapper like for Diagnostics to create new models and folders. Read in inputs from examples/inputs\_temp.csv and copy into the folder’s inputs folder. Code an option to re-run the model using that input file rather than the inputs\_temp.csv one
* See inputs\_temp.csv for inputs structure – the goal is to have it contain all of Elan’s expanded code’s inputs, for all arms. That way it is very straightforward to see what parameters were used and make changes, unlike how his code is set up right now
  + Some inputs should still just go in the parameters file: baseline survival and stage shift and names for each of the prop\_\* columns, which correspond to the different arms of the trial (let it be flexible). The inputs\_temp.csv file is the data *without* the stage-shift applied, so the prop\* columns are conditional on stage-subgroup groups.
  + How does the paired issue relate to this? Well, within stage-subgroups, people will get assigned a different treatment and thus get a different HR. Then their survival draw should be done in a correlated fashion. So this is fine.
  + Now, to get nsims, we need to carry through the code 4 matrices of size popsize\*nsim. How about a list of 4 data frames? Or, should I attempt to use Leslie’s framework of using recreate\_data whatever? No, I think we could use that for age, sex and other independent covariates that we may never use, but let’s keep these 4 matrices no matter what and let the other stuff get combined with them
  + So, use Leslie’s code to generate age and sex? Well, she should have really used rmultinom too, so…either way we end up carrying (number of covars) matrices of size (popsize)\*nsim, whether they’re filled with actual values or row numbers.

**9/19/14**: In the 3rd column below, I will list what I need and want for the final code that will produce results for a thoughtful, appropriately simple-yet-useful paper with similarly appropriate easy-to-use-but-flexible code. ☺ Then I will figure out which pieces to lift from teach.

**Older:**

NEW IDEAs (WIP – I’m having trouble figuring out a way to get exactly what I want):

1. Simulate populations of clinically incident cases: age, sex, stage, and other static covars – can include ER+, ER-. Use Leslie’s way?
2. Simulate shifted stage for each sim (matrix of shifted stages)
3. Simulate old treatment for each sim (depends on pop characteristics. Elan’s way is a multinomial draw based on “base.categories” and “base.prop”
4. Simulate new treatment for each sim
5. Simulate four survivals, paired (four matrices, paired)

OLD IDEA (sort of set up):

For run\_file\_screentreat.R:

1. Edit user\_options\_screentreat.R
2. Use wrapper\_screentreat.R to specify version # and run

## Outline of Methods

|  |  |  |  |
| --- | --- | --- | --- |
|  | **elan\_2014-08-29.R**  (Elan Markowitz, Aug 29 2014 – the update in elan\_2014-09-18.R is a great extension of this to include subgroups within the 2 stages, e.g. receptor subgroups) | **run\_file.R**  (adapted from Leslie’s cantrance/screening**)** | **My code (formerly run\_file\_screentreat.R)** |
| Inputs | 1. sim.size = # of subjects in one trial arm 2. advanced.diag.prop = proportion of cases who will be diagnosed in advanced stage 3. screen.RR.advanced = (protective) hazard of being diagnosed as advanced under screening, e.g. 1-% stage-shifted by screening 4. base.mortality.rate.early = lambda for early-stage exponential survival 5. base.mortality.rate.advanced = lambda for advanced-stage survival 6. rx.effect.early = (protective) hazard of treatment on early-stage survival 7. rx.effect.advanced = (protective) hazard of treatment on advanced-stage survival 8. p.rx.old.early = proportion of early-stage cases treated in old trial 9. p.rx.new.early = proportion of early-stage cases treated in new trial 10. p.rx.old.advanced = proportion of advanced-stage cases treated in old trial 11. p.rx.new.advanced = proportion of advanced-stage cases treated in new trial 12. incidence.table = SEER-originated rate of clinical diagnoses per 100,000 by midpoint of 5-year age groups | 1. userdat – either from individual-level data or covariate proportions    1. Age, sex and stage at clinical dx | * Multiple sims of a cohort of size N – the multiple sims allow there to be variation in the covariates we assign independently, e.g. age and stage, as well as random variation in who gets what OCD draw, disease-specific survival draw, etc. As covariates increase (e.g. ER+,ER-,HER2 status), I think this will become increasingly valuable * Elan’s inputs, basically, but with “receptor” as a general “subgroup,” and condensed into 2 or so data frames. Maybe even 1. That way the input could just be a .csv file if I wanted, but could also be just controlled by R and thus easier to set up different runs   + sim.size   + screen.RR.advanced     - 1-stage shift   + Prop.matrix   + Hazard.matrix   + Screen.RR.advanced   + Baseline.hazard.trt1     - Baseline hazards for subgroup \* stage * Consider setting up treatment distributions and efficacies for all 4 arms in the same data frame. Should I make this flexible to the # of arms? –Sure, why not? * An additional age distribution input. This way we are not forced to have everyone age=50 |
| Simulate population of clinically incident cases (sex, age, stage) | **Input data**  Transform incidence.table into incidence.data, with vector incidence.free.survival as cumulative net probability of being incidence-free at single-year ages  **Process**  Use generate\_clinical\_diag to   1. Generate uniform draw for everyone btwn 0 and max incidence-free survival 2. Use to determine age at incidence 3. Set age at incidence to 10,000 if the draw is beyond the max age of the data   **Change**: This process is not linked between arms within trials or old vs new trials  Use generate\_stage\_at\_diag to   1. Generate stage at diagnosis (advanced or early) based on advanced.diag.prop   **Maybe change**: There is no correlation between age at diagnosis and stage at diagnosis. This does not seem problematic in this controlled cohort where all women begin screening at the same age. | **Input data**  Uniformly assign single-year ages if age is specified as age groups.  **Process**  Extract age at clinical diagnosis from the input population  **Change**: Separate the clinical incidence data from the age of the simulated population—they can be the same, or they can be different. How: for the latter, require an incidence\_table with agegroup and rate of incidence, and let categorical\_chars be assigned in an integrated fashion: first age, then age at incidence (let draws be based on entry age), then everything else | Simulate age at clinical incidence, taking into account current age – easy peasy. Very similar to the life table |
| Simulate treatment in the population | **Process**  Simulate treatment among early-stage cases using p.rx.early and among advanced using p.rx.advanced. Assign baseline mortality rates according to stage and apply treatment hazard ratios according to treatment to get final exponential mortality lambdas for each person (4 total possible: early/advanced and treated/untreated combos).  **Change:**   1. This process is not linked between trials 2. There is only one treatment hazard ratio allowed per stage | **Process**  Specify treatment as one of the categorical\_chars or in the individual-level data. Seems like it is easy to specify a joint distribution of stage and tx with as many categories of tx as desired. | Use Elan’s process, but use sim\_same\_qexp (allow to-be-coded generalizing to empirical survival curves) |
| Simulate screening outcomes for population | **Process**  Use the same process as for the unscreened population, except that the proportion of cases diagnosed in advanced stage is now lowered from *advanced.diag.prop* to *advanced.diag.prop\*screen.RR.advanced*  **Change:** This process is not linked to the unscreened arm | **Process**  In scr\_stage\_dist, specify an order to the stages and a distribution in the presence of screening. Encompasses a proportion shifted from advanced to local, but generalizes to more stages.  The shifted stage is bootstrapped from the original stage using bootstrap\_shifted\_stage  **Note**: I’m unclear how this works separately for individual\_data vs covariate\_proportions. I should look into bootstrap\_shifted\_stage more | Use Elan’s process, but linked somehow – maybe, specify which arms get a stage shift and use the SAME stage shift. So there would only be 2 stage matrices, shifted and not shifted, and then start with treatment in the unshifted stages, and then move to treatment in the shifted stages and only change treatment for the shifted cases. I think Elan did this. |
| Simulate time from clinical incidence to cause-specific death in absence of screening | **Process**  Simulate from exponential curve using the rate determined by the combination of baseline rate and treatment hazard | **Process**  In mort\_param and mort\_covar1…etc, specify baseline survivals and HRs by any covariates present in the simulated population | I like the Cantrance framework because you can throw in age, etc – ANY covariate present in the data can stratify the baseline rates or add a hazard. This could be useful for the PC example—I just don’t know |
| Simulate time from clinical incidence to cause-specific death in presence of screening | **Process**  Simulate from exponential curve using the rate determined by the combination of baseline rate and treatment hazard. The distribution of rates will be shifted towards smaller rates due to more cases being in early stage.  **Change**: This process is not linked between arms | **Process**  In mort\_param and mort\_covar1…etc, specify baseline survivals and HRs by any covariates present in the simulated population | Do this 4 times, but paired: use sim\_same\_qexp or maybe sim\_same\_qdistr, where empirical could be an option |
| Simulate time from clinical incidence to other-cause death | **Input life table**  NCHS, year ?  **Process**  Simulate an age at other-cause death that is between the min and max ages of the input life.table. This is a simplified process because it assumes that all individuals enter at the same age, the minimum age in the life table. | **Input life table**  BMD  **Process**  Standard CANTRANce  **Change:** allow for NCHS life table just like I did for ODX | Make it flexible to the life table, like I did for Diagnostics |
| Difference between old trial and new trial | Captured by the difference in p.rx.new.early vs p.rx.old.early and p.rx.old.advanced vs p.rx.new.advanced. This means that the new trial can have different proportions of treated cases by stage. | **Process**  Would have to do two runs and compare the outputs  **Change:** embed the 2nd trial as an option within the code, so that everything can be paired. I think, stick to treatment being the only difference between the trials. I’m ok having to rework the code if there comes a time when we want to compare different stage shifts or different anything else. | It’s now a multi-arm trial. I need a mock-up.  Compare the two trials (MRR, NNT) but also compare the screening arms and the no-screening arms:   * MRR\_screen\_oldtreat * MRR\_screen\_newtreat * MRR\_treat\_screen * MRR\_treat\_noscreen |
| Outcomes | time = Time of death is the minimum of *ocd* and *survival*, the respective ages at death from other causes and cancer  event = 1 if cancer death, 0 if other-cause death  create\_survival\_table tallies cumulative incidence cancer deaths every 5 years | delay\_summ – time by which screening delays cancer death?  **Change**: all outputs would need to be restructured to accommodate 2 trials. Or, could I conceptualize the code as a multi-arm trial? I think it will end up being the same. | What might I do different runs for?   * + - Different stage shifts     - Different treatment efficacies by stage   I think the easiest way to keep the code manageable is to have a wrapper script that loops, like my PSA script for Diagnostics, and inserts parameters. Then it collates outcomes across runs, produces graphs, etc.  This means that the main script should be able to receive parameters from an inputs script. Hmmm. So instead of data table inputs, the data tables should be constructed of vectors that can change based on a inputs script that could have loops of vectors and be sending just one new row/column.  Well actually, it’s fairly easy to set up a loop that changes one row or column of a DF.  Could set up all the vectors at the beginning of the script, and then have a run-script at the end that could easy incorporate a loop |