A retirement planner and withdrawal optimization model

Introduction:

What is the problem we are concerned with here?

For most of us the amount we are able to build up for our retirement is less than what we would like. This creates challenges for us in our retired years. In order to make the most of what we are able to do with these funds we look to our financial advisors. The financial community has understood for a long time that the specific withdrawal patterns can make a significant impact on the amount available to us and how long our funds will last.

One of the first articles I read[[1]](#footnote-1) along these lines discussed 15 separate withdrawal strategies, chief among them is the Common Rule (CR) and its comparison with the winner, starting with withdrawals from a Tax Deferred account up to your level of deductions first (TDD) strategy variant. The common rule, simply stated, is to withdraw from after tax accounts first, then the tax deferred accounts and finally the tax free accounts. The idea is to allow as much growth as possible in accounts sheltering that growth. TDD on the other hand shuffles the order to start with some portion of the tax deferred accounts, followed by after tax accounts, next the tax free accounts and finally the rest from the tax deferred accounts. Here the idea is to withdraw the amount equal to the level of deductions from the tax deferred account effectively withdrawing taxable money tax free. Additionally this lowers the amount that will have to be withdrawn starting at age 70 ½ for the Required Minimum Distribution (RMD) required by the IRS, which may prevent some creeping into higher tax brackets caused by the RMD. The paper finds TDD outperforms CR by $400K more remaining and paying $225K less taxes when starting with $2M (IRA 70%, After Tax Account 20%, Roth 10%) and a $50K yearly withdrawal.

A primary tool for their work as discussed in their paper was a Linear Programming (LP) model that fully accounted for tax implications. However, they do not go into the details of the model. Thankfully, there is a substantial example[[2]](#footnote-2) from 1994 that can serve as a guide for those of us not completely comfortable with LP models.

The main focus in the above work is on where to withdraw funds for each period of retirement in order to get the most from what you have. Another line of thinking that is even more important is focused on how much to withdraw while ensuring your funds do not run out. The 4% rule[[3]](#footnote-3) seems to have been the first systematic approach to this line of thinking and is well worth the read. The basic idea is to withdraw 4% of your funds the first year of retirement. Each subsequent year, increase the 4% by the previous year’s inflation rate. In many cases (dare I say most) following this advice will leave a substantial amount for the estate. In a few cases (have some examples for this) however it may still fail to fund the retirement in its entirety.

Looking to improve the guidance for where to make withdrawals while also ensuring funds will last throughout retirement has pushed LP models forward to be use for retirement planning and year to year execution as with 3-PEAT.[[4]](#footnote-4) ---Now what to say ---

(Give refs and examples…) and a later ref

Current environment and the rules that govern it:

Discuss each of the laws as Ragsdale does (about a paragraph or so each)

* Contribution (We ignore company retirement contributions…)
* Early withdrawal
* Minimum distribution
* Tax method
* Estate tax

Now, what about the model and its mathematical representation?

A little background may be required at this point. Linear Programming (LP) is a mathematical technique for optimization. The name was chosen prior to the wide spread use and development of software and the programming involved to create it. Rather, when using LP we define an object function to be optimized, a set of real valued variables to be determined, and a set of constraint expressions that set requirements on the optimal solution. This gives us everything we need to set up and solve our matrix expression:

(0)

Where x is our vector of variables, each row of A is represents a constraint such that when this row is multiplied by x the corresponding element of b appears on the Right Hand Side (RHS) of the expression. Once the problem has been defined in this manor, expression (0) can be solved for the optimal value of the object function (variable vector) c, ct is the transpose of c. With that out of the way, lets move on.

OK, so we are looking at a new object function as well as the original we got from Ragsdale. In the Ragsdale model the overall effect of the mode was to optimize for the largest Estate possible. To get it to work on optimizing the way funds are removed from the various account it needs a desired spending amount. Without this, the model will not remove any funds from any account that is not required, like the RMD (Require Minimum Distribution). If we are not inclined to maximize our final estate, but would rather maximize the funds we can spend each year we need to emphasize the yearly spending component for the object function. With such an object function we might want to specify a maximum spending level, above which we’d rather keep the funds in the estate. Let’s start with the new (experimental) object function (S1) and then we will rewrite the Ragsdale object function as a variant (R1):

(S1)

(R1)

The idea here is that we want to maximize the spendable (si) dollars across the retirement years. For Ragsdale we also want to maximize the remaining account balances. However, we don’t want to require that all the available account balances is use up in cases where we set a maximum spending amount. So, in both the cases we define the object function to include both the sum of the yearly spendable amount and the sum of the account balances at the end of the period times a discount rate for each account (DjA). For (S1) though we include a “balancer” to lower the significance of the final balances in the optimization such that spendable funds will be favored. The balancer is using a heuristic of dividing by the sum of the initial account balances. The account discount rate is applied to the final balances to suggest the value of the balance given how the account is taxed; TDRA at 0.85, ROTH at 1.0, and the after tax investment account at 0.9.

The spendable amount (si) is the sum of all withdrawals (wij) plus Social Security (SSi) and other income (oi) minus special expenses (ei), income tax (xik subject to tk) and capital gains tax (yil subject to tl) as well as money deposited back into the investment account (Di). The following constraint is used to assign the spendable amount to si.

(2)

(2 old)

In general we want allowable spending to increase with inflation and to remain steady year to year so we have the following constrain:

(3)

The objective function (R1) consists of two present value components. The first component calculates the sum total present value of the taxpayer’s after tax income for n years. The second component calculates the sum balance of funds remaining in all accounts in nl, one account per person of each type of the tax deferred accounts and one after tax account, in the n+1th year. The account types include:

1. n1, Tax Deferred Retirement Accounts (TDRA) including 401(k), traditional IRA and similar plans (one per person),
2. n2, Roth Retirement Accounts (RothRA) including after tax contributions to 401(k) and similar as well as Roth IRAs (one per person) and
3. n3, After Tax Retirement Savings/Investing Accounts (ATRSI) (shared).

Thus, the objective specified by expression (S1) attempts to maximize the withdrawals and other taxable income (after tax) made over the taxpayer’s retirement period. Similarly, (R1) does the same while maximizing the remaining account balances.

Note: This model uses one account for each type of account for each person such that the balance and withdrawals for the model account represent the sum of the balances / withdrawals for any number of accounts of that type. One sticky point here is when, for example, a 401(k) contains both tax deferred and after tax contributions. In this case the balance must be split and added to the correct account types.

The tax rates (tk) represent the rate for ordinary income for each inflation adjusted tax bracket while tlcg is the capital gains tax rate for each capital gains inflation adjusted tax bracket. Currently, though, tlcg is being modeled as a single bracket with one rate (15%). (TODO: this paragraph is out of date, it refers to (2 old) and not (2))

Questions to address: Maximizing the after tax yearly income and the TDRA balances at the end may have issues, is this really what I want? Maybe maximize include while keeping estate a certain level? Or meeting an include amount while maximizing the estate? How to handle the estate taxes. I’ve dropped them in the above.

(4)

(5)

A number of constraints are required to constrain the objective function in expression (R1) and (S1) to optimal values while ensuring that IRS rules are followed and taxes and penalties are properly accounted for. Additional constrains will be given to allow for a minimum level of income per year. To this end we define expression (4) to ensure that each year’s spendable amount (si) is at least as much as the desired income (di) for every year in the modeled retirement period.

(6)

Of the three account types, only the TDRA has an IRS requirement for a Minimum Require Distribution (MRD). This requirement applies to all such accounts but the sum of the MRD can be withdrawn from each account, any one of the accounts or some combination of TDRA accounts as long as the full amount of the MRD is withdrawn from TDRA accounts. To ensure withdrawals are at least as much as the IRS minimum required, equation (6), requires withdrawals, starting at age 70 (n70), exceed the balance in the TDRA (bi1) divided by an IRS defined life expectancy value (aij this should include a way to choose which expectancy table is needed).

(7)

(8)

Equation (7) constrains the variable representing the amount of income in tax bracket k (xik) to take on an amount related to the total taxable income (the TDRA withdrawals and other taxable income, oti) minus the deductions (standard deduction and exemptions, sdi). Expression (2) forces income into the lowest possible brackets through the applied tax and need to be maximized. That is, tk is monotonically increasing as k increases in expression (2), which forces the xik in the lowest brackets to fill first. Expression (8) ensures that the xik portion of the income does not exceed the bracket amount (mik). Mik and sdi are inflation adjusted.

(9)

(10)

In the same manor (9, 10) fill the capital gains tax brackets (yil) with the non-basis portion (pi) of the investment account (ATRSI) withdrawal (winl). However, the capital gains tax bracket fill must start where the ordinary income bracket fill stopped and continue up from there. In order to do this we add another constraint on yil that may lower the bracket size by making it less than or equal to the floor of the next bracket level (Fl+1cg) minus the total ordinary taxable income (wi1 + oi +SStSSi-sdi). Or as in the alternative constraint (11 Alt) we simply subtract the sum of the ordinary taxable income. One problem with this is it may force yil into negative territory which would cause the model to have no solution so we add nsil to take up any negative amount that the constraint causes. (TODO: this paragraph is out of date)

(11)

(12old)

Equation (12) ensures that the balance for each account at the beginning of the year (bi+1,j) is equal to the balance of the account at the start of the previous year (bij), minus the previous year’s withdrawals (wij) (modeled as being withdrawn at the beginning of the year) times the return on the investment for the year as rate of return (rij). This is somewhat pessimistic because withdrawals are usually not taken out in one transaction at the beginning of the year but this is a small effect for our purposes here.

(13)

Expression (13) sets the beginning account balances to qj.

(14)

Finally, expression (14) constrains the model variables to be greater than or equal zero.

Transforming our model into a python implementation:

OK for our current work we will use the python scipy library, specifically the function scipy.optimize.linprog(). This requires the model to conform to the following template:

Object function: Minimize ct x

With constraints: A x <= b, and x >= 0

(need to zero base the indices)

Given this we transform our model expressions to match scipy template form as follows:

OK, as above we will first look at the new objective function followed by the Ragsdale objective function.

(S1’)

(R1’)

(2’)

Pij == 0.1 for i <60,j<nl Otherwise pij ==0

(2’ old)

(3a’)

(3b’)

(4’)

(5’)

(5+’)

(5++’)

Deposits to the IRA and roth accounts must not exceed any of these three: other income (oi), IRS define maximum contribution (mci) or user contribution level (uci). MAYBE: can add a specific variable for IRA to Roth conversions and not consider it to be done with deposits (Dij).

(6’)

(7a’)

(7b’)

(8’)

(9a’)

(9b’)

(10’)

(11a’)

(11b’)

(12a’old)

(12b’old)

Bij supports an extra year

(13a’)

(13b’)

(14’)

To transform our model into the scipy equivalent we have only to use a few operations. To transform the object function, expression (R1’) or (S1’), we minimize the opposite of (R1) or (S1). The rest of the expressions are transformed by multiplying by minus one (-1) to change greater than or equal (≥) to less than or equal (≤), moving constants to the Right Hand Side (RHS) of the expression and doubling up equations to convert from equal (=) into two relations, one with greater than or equal (≤) and the other with less than or equal (≥) (properly transformed) to bring all constraints into standard form (i.e., A x ≤ b). In (11a’ and 11b’) we also multiplied out the c(bij – wij) to (cbij – cwij) to more closely match the matrix coding. Constraint (13a’ and 13b’) set the initial account balances to qj.

Model input specification:

The basis for the model input specification is the toml specification[[5]](#footnote-5) with its library. The basic format is sections of information ‘[‘ section name ‘]’ where section name can be a category followed by a ‘.’ and a dispriptive name. ‘#’ to the end of the line represents a comment. The global section has no section name header. The input information is represented by an assignment.

General model information (global section):

retirement\_type = 'joint' # defaults to joint, could be single, joint (married), ??? TODO

returns = 6 # defaults to 6.0, return rate of investments as a percent

inflation = 2.5 # defaults to 0, yearly inflation rate as a percent

#maximize = ‘PlusEstate’ # defaults to ‘Spending’, ‘PlusEstate’ maximizes the final estate.

# individual retiree info (one section per retiree, at least one will need an id)

[iam] or [iam.id] where id is some string that represents the retiree

age = 60 # retiree’s age

retire = 65 # age retiree will retire

through = 95 # age through which to plan

primary = true # if there are more than one retiree one should be designated as primary

# Social Security section must specify amount, FRA and an age range

[SocialSecurity] or [SocialSecurity.id] # if more than one retiree at least one needs an id (id should match retiree id)

amount = 31000 # $31,000 at Full Retirement Age (FRA); Assumes inflation, 85% taxed

FRA = 67

age = "68-"

# Income must specify amount (yearly), age range, whether to adjust for inflation, and if it is taxed

[income.mytaxfree] # income after retirement that does not involve retirement accounts

amount = 3000

age = "67-" # starts at age 67 and continue

inflation = false # Adjust for inflation

tax = false # count this as ordinary taxable income (true/false)

[income.rental\_1] # another income source

amount = 36000

age = "67-"

inflation = true

tax = true # count this as income tax

[income.rental\_2] # a third income source

amount = 2400

age = "67-"

inflation = true

tax = true # count this as income tax

# desire income section must define amount, age range, inflation adjustment and whether it is taxed

# desired income should be use in conjunction with ‘PlusEstate’

#[desired.income]

#amount = 45000 # per year

#age = "68-"

#inflation = true

#tax = true # count this as income tax

# max income section must define amount, age range, inflation adjusted

#[max.income]

#amount = 150000

#age = "68-"

#inflation = true

## sections for each account type, IRA and Roth types should have on per retiree as needed

## only one pre-tax account

## Each accounts must define the initial balance. Optionally, an investment return rate can be given

[IRA] or [IRA.id] # where id must match the retiree’s id who is the owner of the account

bal = 2000000

#rate = 7.25

maxcontrib = 00

# roth type accounts

[roth.spouse]

bal = 100000

maxcontrib = 000

# after tax savings/investment type accounts

[aftertax] # assumes joint ownership if multiple retirees

bal = 700000

basis = 400000 # for capital gains tax

Expression Key:

aij IRA life expectancy at age in year i for account j

Bcg Number of capital gains tax brackets

Bt Number of tax brackets

bij balance of account j in year i

di desired minimal before tax income

dmi desire maximal before tax income

Di deposits to investment account in year i (this may be expanded to other accounts pre retirement?)

DjA The discount rate for the jth account

ei Special expenses that have a limited duration. These will not be counted as part of si

fi the capital gains fraction of investments (i.e., fraction that does not include the basis)

Flcg floor of the capital gains bracket l

i index for number of retirement years

inf inflation rate

j index for the number of accounts

k index for the tax brackets

l index for capital gains tax brackets

mik size of the kth tax bracket in year i

mcgil size of the lth capital gains bracket in year i

n Number of retirement years

n70,j year number that retiree owning the jth account is age 70 (i.e., primary retiree age 70 plus partner retiree age difference)

nl Number of accounts (i.e., number TDRA + number Roth+ aftertax accounts)

nAT Aftertax account index (when there is an aftertax account, nAT will equal nl)

oi Other income in the ith year

oti Other taxable income in the ith year (oti is a subset of oi)

pi the penalty cost of accessing a retirement account prior to age 60 (59½), Age <60 10% else 0%

pvi Present Value in year i

qj balance for account j at the start of retirement

rij rate of return for account j in year i

rcg capital gains tax rate (temp until cg tax brackets are working)

sdi Standard deduction in year i

si Spendable amount in year i

SSi Social Security income in year i

SSt Social Security faction that is taxable

tlcg marginal capital gains tax rate in bracket l

tk marginal tax rate in tax bracket k

wij withdrawal from account j in year i (j=1 TDRA, j=2 Roth, j=3 Investment)

xik ordinary taxable income in year i and bracket k

yil capital gains income in year i and bracket l

Consolidated Todos:

1. Create a case to use Aeq x == beq as well as Aub x <= bub
2. Add to this document a note in the proper place that states that inflation adjusted values assume the years of adjustment equal the number of years from start of retirement.
   1. So income or expense that is to be adjusted should be based on values at the time of retirement (e.g., enter income amount of 100 per year during retirement, will use 100 the first year of retirement, 100\*(1+inflation) for the second year or 100\*(1+inflation)^#yearsRetired )
   2. SS is an exception in that it is adjusted twice, once for the IRS adjustment depending on the start age vs. FRA and then for inflation as above.
   3. NEED TO ENSURE SS WORKS correct if it has already started (i.e., we are already retired)
3. Add to this document a note that income and expense info timelines tie back to the primary age line. That is, if [income.apt] include ‘62-‘ then the model will match this with the primary retiree’s 62 year and forward.

To Add:

* Experiment: what happens if I use 2x si rather than 1x ?
* MUST FIX. Like to enable the user to plug in a value for the standard deduction + exemptions.
* Should I assume any interest yields??? Currently it’s just capital gains
* Other taxes (Medicare tax, ???)
* What about fsic (social security tax?)
* Another problem with this is it DOES NOT ALLOW FOR A CHANGE IN THE BASIS while optimization is happening. Why is this important? Because, I want to be able to have excess withdrawals placed in ATRSI (bi3) which would need a corresponding change to the basis bmi this would require it to be a variable (not a constant) but I don’t think this method allows for variable to be multiplied. NEED TO VERIFY

To Do:

* Break out the consistency checking code into a separate file
* Make all output headings consistent and useful (uniform across the different ones (Taxable done the same everywhere))
* Convert the indices to zero based to match the python code
* Add selectable MRD tables (single, joint married, ???)
* Add state taxes
* ~~Add for retirement pre age 60 (TDRA 10% penalties, …)~~
  + Add a check box to not use this is an exception applies (add an attribute in the accountable?)
* Can the tax brackets be changed somehow to significantly lower the number of variables?
  + Want smaller / faster model
* Add checks to eliminate constraints where not needed
* New Auld model:
  + Add pre-retirement, how much to add to tIRA, tRoth, tSavings, tInvestments (use wayne scott or james welch jr style pre retirement; ws include in LP jw I don’t think does)
    - Add an output row for the current state (balances…) followed by the retirement rows, such as:
      * Current time row (input values)
      * Start row input values modified to match expected values at retirement
      * Year to year rows
      * End row with final state of the values
  + Input to optimize:
    - Add fRoth to the tax summary page in case of a 10% early withdrawal penalty
    - add a maximum spend rate (to for excess withdrawals into Roth accounts
    - ~~I think I will need to add two TDRA and two ROTH accounts to correctly handle joint where both have these accounts (RMD differ, timelines differ, SS differs)~~
    - Think about: if want to enable the model to deposit money in any of the account types the it has to be present whether or not its in the input file. Given this we could attach it to the primary retiree
      * Might help answer the question on whether a spouse should continue to contribute to IRA or Roth after another spouse retires
        + To add to the IRA you need to have taxable income of equal amount (not sure of the restriction for Roth)
    - Add version function and information (tag a commit for specific spots)
    - What mix for Investment account? (100% stock, stock bond mix?)
      * Capital gains taxes vs. interest vs. income tax
      * Currently assuming 100% -- But this is not most people or standard advice
    - Other sources of income include
      * illiquid assets (house)
        + will require an investment account to deposit the funds
      * reverse morgage
      * yield (is this interest? From?)
    - Like to optimize start year of SS
    - Short term savings goals (car, ….)
      * ORP use a list of time x amount tuples
      * Fplan has an expense vector from input (implementing this! Has model stability problems)
    - Emergency fund (6months – 1year, family emergencies,…)
    - Run for time prior to retirement
    - Break out SS husband and wife in print out
* Ability to run in simulation mode against a defined return rate for each year (ie., a portion of the historical S&P 500 record).
  + Compare against other strategies:
    - 4% initial amount forever, or with inflation, or adjust to 4% each year[[6]](#footnote-6) [[7]](#footnote-7) [[8]](#footnote-8)
    - CR
    - TDD
    - Autopilot (<http://www.marketwatch.com/story/theres-a-better-way-to-plan-retirement-withdrawals-2015-01-13?page=2> )
    - RMD all the way
* Like to be able to consider two peoples accounts more fully. That is to be able to determine things like:
  + When older retires and is withdrawing from TDRA does it make sense for the younger to continue to add money to the TDRA, maybe even enough money for the limit of both.

1. Sumutka, Alan R., Andrew M Sumutka, and Lewis W. Coppersmith. (2012). Tax Efficient Retirement Withdrawal Planning Using a Comprehensive Tax Model. Journal of Financial Planning; April, Vol. 25, Issue 4. Retrieved from: <https://www.onefpa.org/journal/Pages/Tax-Efficient%20Retirement%20Withdrawal%20Planning%20Using%20a%20Comprehensive%20Tax%20Model.aspx> [↑](#footnote-ref-1)
2. Ragsdale, Cliff T., Andrew F. Seila, and Philip L. Little. 1994. “An Optimization Model for Scheduling Withdrawals from Tax-Deferred Retirement Accounts.” Financial Services Review 3, 2: 93–108. Retrieved from [www.twenty-first.com/pdf/RagsdaleAn\_Opt\_Model.pdf](http://www.twenty-first.com/pdf/RagsdaleAn_Opt_Model.pdf) [↑](#footnote-ref-2)
3. Bengen, William P (October 1994) Determining withdrawal rates using historical Data. Journal of Financial Planning Pages 171-180, Retrieved from: <http://www.retailinvestor.org/pdf/Bengen1.pdf> [↑](#footnote-ref-3)
4. Welch, James S Jr. A 3-Step Procedure for computing sustainable retirement savings withdrawals. Journal of Financial Planning 30 (8): 45-55. Retrieved from: <https://www.onefpa.org/journal/Pages/AUG17-A-3-Step-Procedure-for-Computing-Sustainable-Retirement-Savings-Withdrawals.aspx> [↑](#footnote-ref-4)
5. The toml specification is archived at: <https://github.com/toml-lang/toml> [↑](#footnote-ref-5)
6. Bengen, William P (October 1994) Determining withdrawal rates using historical Data. Journal of Financial Planning Pages 171-180, Retrieved from: <http://www.retailinvestor.org/pdf/Bengen1.pdf> [↑](#footnote-ref-6)
7. Bengen, William P (May 1, 2012) How much is enough? Financial Advisor, Retrieved from: <http://www.fa-mag.com/news/how-much-is-enough-10496.html> [↑](#footnote-ref-7)
8. Bengen, William P (June 1, 2016) Is 4.5% Still Safe? Financial Advisor, Retrieved from: <http://www.fa-mag.com/news/is-4-5---still-safe-27153.html> [↑](#footnote-ref-8)