Retirement Optimizer User’s Guide[[1]](#footnote-1)

(Includes new 2018 tax code)

Introduction:

In this document we will describe the objectives for using the Retirement Optimizer (RO), some simple cases and how tos. For this there is a section on the model input specification as well as a section on the model’s output.

While not the objective for this document it is good to know that this tool is to provide an optimal plan for retirement that, in many cases, can be worth tens of, or even, hundreds of thousands of dollars. The financial community has understood for a long time that our specific withdrawal patterns can make a significant impact on the amount available to us and how long our funds will last. Four papers that are worth your time are:

1. Tax Efficient Retirement Withdrawal Planning Using a Comprehensive Tax Model[[2]](#footnote-2)
2. An Optimization Model for Scheduling Withdrawals from Tax-Deferred Retirement Accounts[[3]](#footnote-3)
3. Determining withdrawal rates using historical Data[[4]](#footnote-4)
4. A 3-Step Procedure for computing sustainable retirement savings withdrawals[[5]](#footnote-5)

The Retirement Optimizer is based on the results of these and similar papers.

The Retirement Optimizer is aimed at helping retirees to get the most out of what they have. This does not, for the most part, include how to optimize your actions in getting to the point of retirement. That said, there are at least two legitimate uses that occur prior to your actual retirement:

1. Using the tool to assess whether you can retire and planning near term retirement
2. Using the tool for a married couple where they do not retire the same year

In these cases there is a period of time where not everyone (or no one) is retired.

To use this application, the first step is to prepare your input information. The information can be defined in either of two file types: toml or strmap. The package includes some example toml files including one to be used as a starting point. The input configuration defined in the toml file is used to set up the constraint model to represent your specific situation for the solver to produce a proposed plan of action. The plan of action consists of a list of withdrawals (and deposits) that optimize your use of your retirement funds. The model accounts for all your federal income and capital gains taxes, rules for handling tax deferred retirement accounts, social security and other incomes as well as the sale of major assets like a home.

The current implementation, applies the rate of return for each account to the given balance between the current year and the start year for the retirement plan (year of the first person to retire). Also, contributions for each account that occur during this pre-plan period will be added each year and will then also receive growth at the account’s rate of return. However, no checking is being applied in the pre-plan period for meeting IRS rules. This, of course is not ideal. Once in the plan period (first retiree retire age through last retiree’s plan through age) contributions must adhere to IRS rules. As such, to take advantage of allowable contribution levels for those participating in an employer sponsored defined contribution plans you must specify your participation period (start through end age).

Accounts are modeled as having a yearly balance, deposits and withdrawals. Each retiree can have a Tax Deferred Retirement Account (TDRA) and / or a Roth Retirement Account (RothRA). Additionally one After Tax Retirement Savings / Investment account (ATRSI) can be included. So a single retiree could have up to three accounts being modeled (one of each type) while a married filing jointly could have up to five accounts (one per retiree for the retirement accounts and one non-RA or investment account).

To model your accounts you would sum your balances for all your accounts that act as a traditional IRA (Traditional IRA, Traditional 401(k), 403(b), 457(b)) into the TDRA (IRA in the toml file) account starting balance. Then do the same for all your Roth account types. This should include all Roth deposits in a 401(k) if any. And finally, sum all the accounts with no special tax treatment for retirement for the after tax account.

For the most part 401(k) should be included in the TDRA account but sometimes they include portions that were contributed with after tax money. In this case the after tax contributions should be included in the RothRA account. IRA account balances go to the TDRA account and Roth IRA balances go to the RothRA account.

Once you have a handle on your account balances for each of the account types, you can choose an inflation rate (defaults to 2.5% per year) and an overall rate of return for investments (defaults to 6% per year), set your retirement type, joint, mseparate or single (defaults to joint) and enter your account information. This should match your tax filing status. The final information you must specify is your current age, age you plan to retire and through what age to create the plan. Note that all age and periods are inclusive. That is, they include each year given. For example, if you will continue to contribute to a 401k at work through to retirement and you want to specify this in the toml file you would specify your current age to start followed by a ‘-‘ and then your age just prior to retirement for the end of the period, [a-b]. With this there is enough for your first run.

Optimize for Spending or PlusEstate: You can choose to optimize for a maximum spending amount during retirement or for the final estate (PlusEstate). This setting defaults to optimize for spending and will create a plan that starts at a value of X dollars a year and increasing this amount for inflation each subsequent year. This gives a good starting point to understand where you are in terms of your ability to retire. If the first year amount is too low then you’ll have to do some serious thinking before you retire. If on the other hand it’s more than you believe you need or want, you can set a maximum spending amount and let the remainder stay in your accounts to increase your estate.

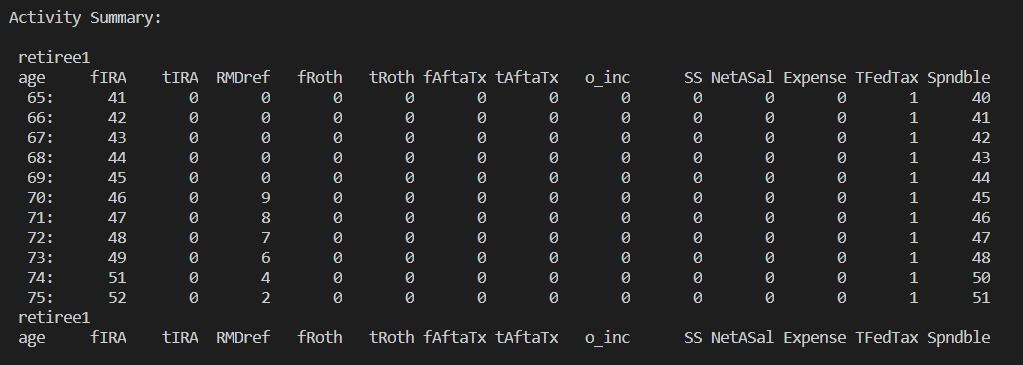
Another approach when you want to maximize your estate and you know what you will need each year, is to optimize for the estate (PlusEstate) and set a minimum income amount. In this case the solver will attempt to provide the minimum amount you specify for the first year of the plan and that amount increased for inflation each subsequent year. The plan will maximize your estate subject to this income.

The iterative process of using the model: Experiment with different input to see how to get the best results. Changing the date of retirement or when to start social security will give different plans and you can choose the one you like best. Higher rate of inflation and lower rate of return will give a more conservative plan. And while the solver creates a plan for the entire retirement planning period it is not intended that you would actually use this one plan for the entirety of your retirement. Over the course of retirement inflation, rates of return, spending needs, tax laws… all change as do your needs. So, each year you should update your information in the model and rerun it. Try some new ‘what if’ runs and see what seems best. Once you settle on a plan with your given specification, follow that plan as much as it make sense to.

Get your feet wet: In this section we will describe a very basic application of the Retirement Optimizer. The first thing that you will need to do is to choose an example input file and modify it to include your specific data. Married and want to do a joint plan, choose ROJointStartHere.toml to start. Single then choose ROSingleStartHere.toml to begin. Copy your choice to whatever you want to call your plan input file, say try.toml. This won’t give a very interesting result when run because almost everything is commented out. Give it a try with:

PS C:\plan> ro.exe try.toml -A

The following table is always printed on success of the optimization. It gives an overall summary of the proposed, optimized, plan. Here we can see our retirees, their age and how much should be withdrawn from the TDRA account each year (fIRA). As can also be seen the spendable amount for the year matches the withdrawal amount minus the taxes; well almost. If no rounding to the nearest thousand were in play this would be true as we will see in a moment. Starting at age 70 the Required Minimum Distribution (RMDref) kicks in but is less than the amount to be withdrawn so it has no effect on the outcome. TDRA is the only account with activity, no other income, social security, expense or taxes so these all remain zero.



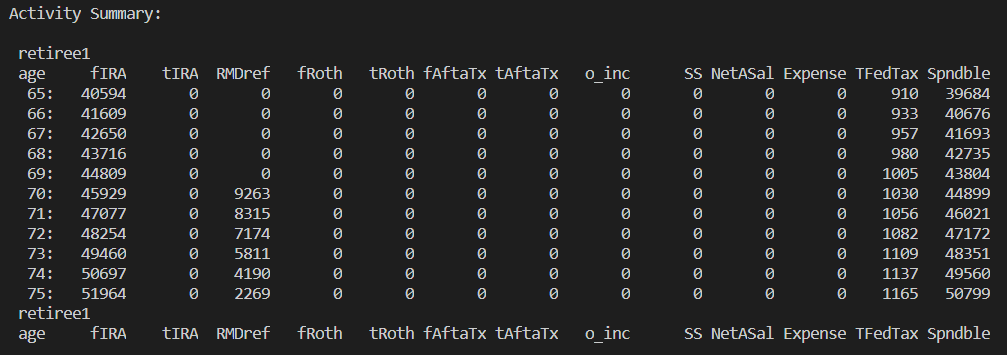
All the numbers are thousands (000) unless the -k switch is given on the command line.

It helps to understand how these values are related within a year. Spendable equals the sum of the amount withdrawn from the accounts minus the amounts deposited, plus other income and social security and any net fund from asset sales (like a house) and finally minus expenses and total federal tax. That is spendable is your after tax money minus expenses. Now in this light, expenses are meant to be major or unusual expenses rather than your normal expenses that you will have over the course of retirement. They are a way of telling the model that an extra amount is needed in certain years.

One way to think about these numbers or to compare them with your situation before retirement is to compare your pre-retirement annual income to the sum of the plan total Federal tax and spendable amounts for the first few years. Beyond this, inflation will overly distort the numbers for comparison. Setting inflation to zero could be used to compare farther out but that is a longer discussion than I want to take on here. Note that this ignores expenses and this is ok if they are not every year events.

The main reason to give a spendable amount rather than an annual income amount is that the annual income product by summing total federal tax and spendable will likely jump around a lot as compared to the spendable amount. This is because the optimization is selecting the best ways for you to consume your funds and this by itself results in very different tax burdens as the source of funds for each year changes.

Before we get to far from the point on rounding to the nearest 1000 mentioned above, let’s see what the same activity table would look like if we had include the –k switch:

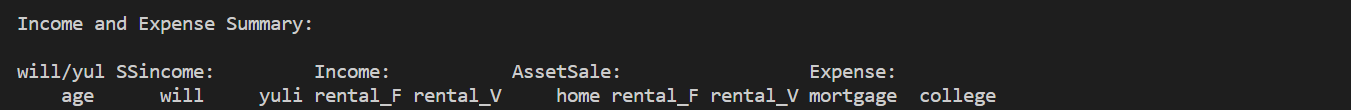


Now from this we can see that the withdrawal minus taxes does indeed equal the spendable amount all rounded to the nearest dollar.

The –i switch given on the command line causes the income and expense summary to be printed as below. Not much to see from try.toml sense it does not define [SocialSecurity], [income], [asset] nor [expense] sections. If these were defined additional columns would be displayed in sections as in the snippet based on a different toml file after our try.toml snippet.



The SSincome section would come first with each column labeled with the retiree id key if defined or ‘SS.’ Next would come the Income section with each column labeled with the income id key. Then the AssetSales section with columns labeled with the asset id key and finally the Expense section with columns labeled with the expense id keys. Note that, as we will see below, the id key value is the text string after the ‘.’ in the toml file for each of these sections. An example for [income] is [income.rental\_F] where ‘rental\_F’ is the id key for one particular [income] section.



The –a switch given on the command line causes the account transaction summary to be printed as here below. Only one account is documented because only one account is specified in try.toml. In addition to the withdrawal (fIRA), deposit (tIRA) and RMD reference (RDMref) this summary includes the balance for the account (IRA). All modeled accounts (those defined in the input file) will be summarized here.

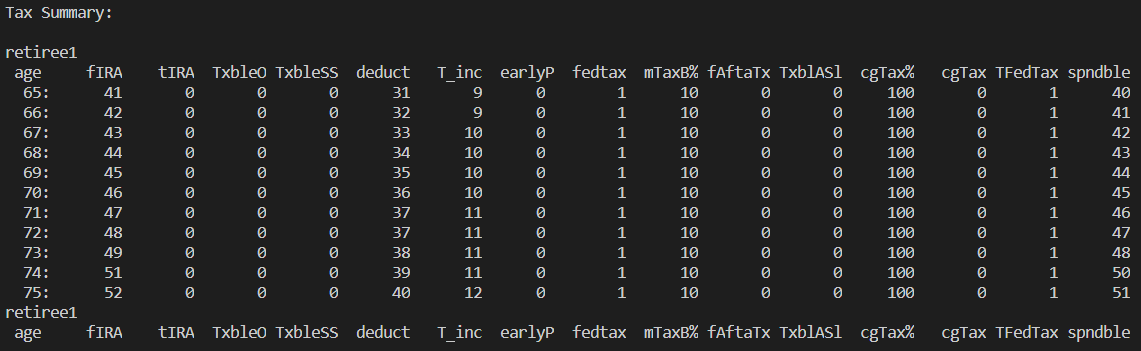
The first data line is for your current year which may be prior to the plan start. It lists the original balance for each account and any contributions specified. The other fields remain zero.



As you can see in the IRA column, the balance jumps from 200K at age 54 to 380K at age 65, plan start. This jump is 200K times the rate of return (6%) to the 11th power. That is the increase is based only on the rate of return for the account over the 11 year period. There are no contributions. They would be specified in the tIRA column. In this model the rate of return includes inflation, that is it is not the ‘real’ rate of return which would not include inflation.

The tax summary is displayed if the –t switch is given on the command line. Here we see the withdrawals from the TDRA account as they are taxed as ordinary income. Also the taxable portions of other income (TxbleO) and 85% of social security (TxbleSS). This is followed by the allowed deduction and exemptions (deduct) and the Total taxable income (T\_inc). Early withdrawal penalties (earlyP) comes next followed by the federal tax amount (fedtax) and the marginal tax rate (mTaxB%). Next comes the withdrawals from ATRSI (fAftaTx) followed by the taxable amount from the sales of assets like a house (TxblASl). Next the fraction of the ATRSI withdrawals that are not from basis, that is the gains on the investment, (cgTax%) and the capital gains tax itself (cgTax). Finally the total federal tax, fedtax plus cgTax, is given (TFedTax) and the amount of spendable funds for the year (spndble).

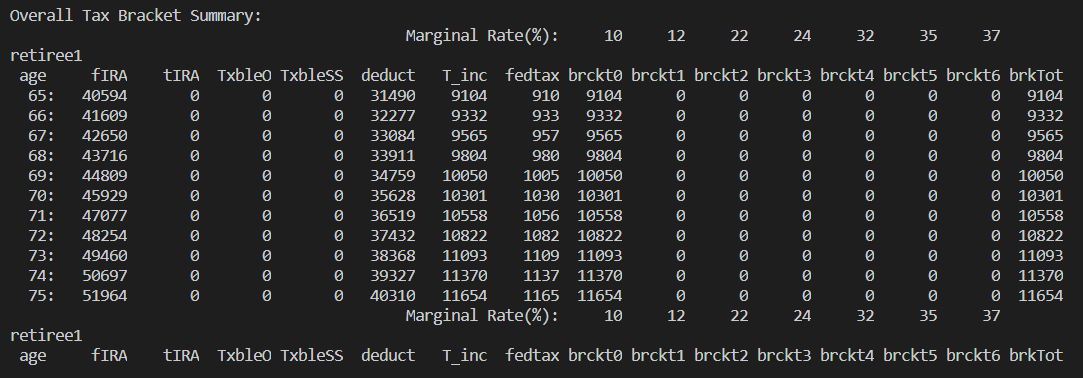
If the early withdrawal penalty includes a penalty for early withdrawal from a Roth account the value will have a ‘\*’ attached. In the current implementation Roth early withdrawal penalties are assigned more in the model than in life. It should rarely happen but when it does this indication will allow you to check to see if it is significant to your plan.

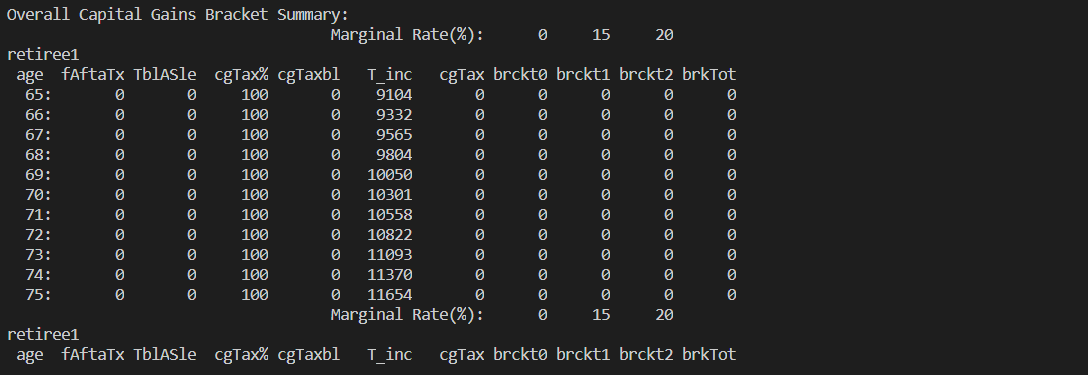


As we saw in the account activity summary it helps to understand how these columns are related. There are three basic subsections: income tax, capital gains tax and and summary. For income tax withdrawal and deposit (fIRA, tIRA), the taxable portion of other income and social security (TxbleO, TxbleSS) and deduction and exemptions (deduct) are all combined to calculate the total taxable income (T\_inc). It is this amount that is used to determine the marginal tax bracket, federal income tax amount (mTaxB%, fedtax). The second section, capital gains, uses withdrawals from the ATRSI investment account times the non-basis fraction of withdrawals to determine the portion (profits) to be taxed (fAftaTx\*cgTax%) plus the taxable amount from any asset sales (TxblASl) to determine the capital gains tax (cgTax). Finally, in the final section these results are brought together to define the total federal tax (TFedTax = earlyP+fedtax+cgTax) with spendable given for reference.

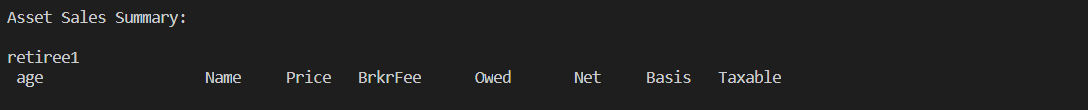
The next three tables are more technical. They are displayed whenever –b is given on the command line. This represents the tax bracket details for ordinary income (first table), capital gains (second table) and asset sales (third table). In the first two tables the new columns are the brackets. Above and below each of these marginal rate columns in the marginal rate itself, the heading (brcktN) where N is a tax bracket from 0 on up to 6 for ordinary income taxes and 0 to 2 for capital gains brackets. The final column is to total of all the funds in the previous column’s brackets for the year. The tax for each year is calculated by summing the amount in each bracket times the bracket marginal rate.

An exception to the rule for all numbers to be in thousands unless the –k switch is used is that the bracket data will always display every dollar independent of the switch, that is, not in the thousands.



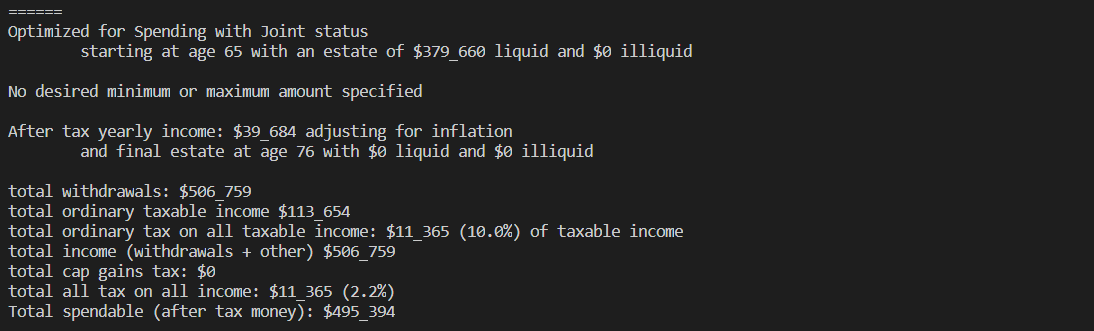


The third table, asset sales, give a summary of information about any asset sales that are scheduled to occur during the plan period. It gives the age at which the sale is scheduled to occur, the name of the asset, the sales price, brokerage fee (BrkrFee), amount owed at time of sale, net proceeds from the sale, basis amount (cost + improvements) and finally the taxable amount from the sale.



The final output is always displayed with the following data. The object function being optimized: Spending or PlusEstate. The plan start estate liquid and illiquid value. Any minimum or maximum spendable amounts desired. The after tax yearly income derived from the optimization plan. This is the first year spendable amount and later years will be adjusted for inflation. This is followed by the plan end estate liquid and illiquid value.

After this some information on the overall plan totals are given to provide a more complete idea of the outcome. This includes total withdrawals from all accounts, total taxable income, total income tax and its percentage of all ordinary taxable income, total income, capital gains tax and then all taxes on all income and its percentage. Finally the total amount of after tax spendable money over the plan.



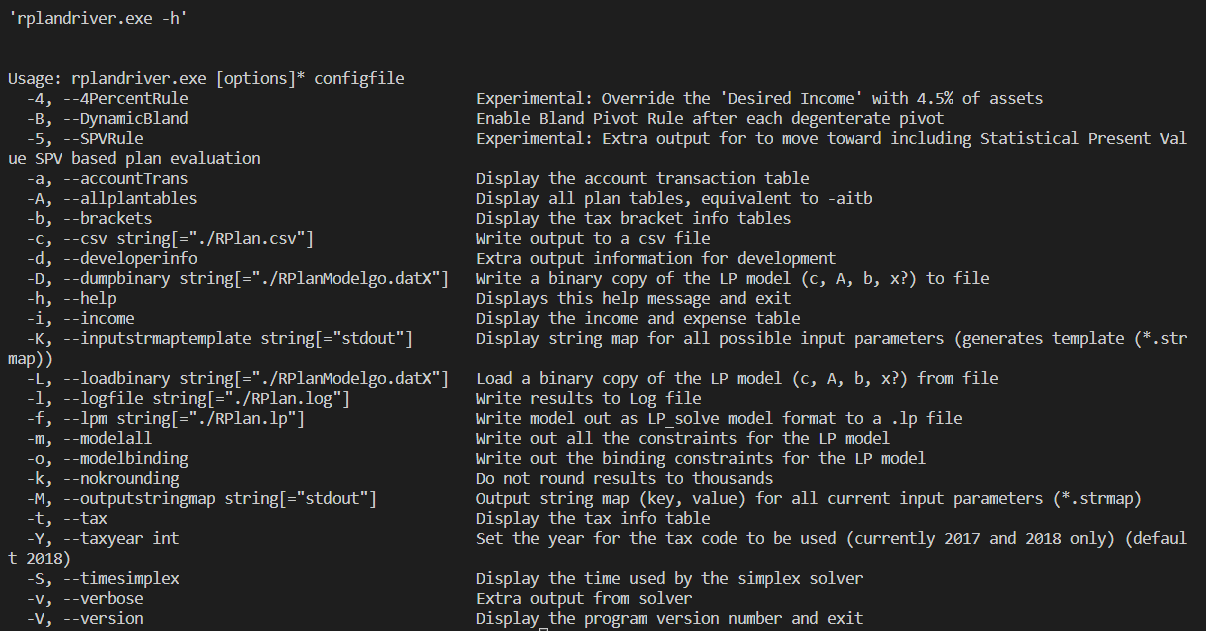
Review the toml file to get a feeling for what is in it. Most options are commented out to simplify your modifications. This way you can change and uncomment only those fields you choose and ignore all the other lines. Once ready modify the file to include your own information. Doing this in small steps and running the optimization at each step will help you understand the impact of your changes and ensure the input file syntax remains correct.

Example input files can be reviewed and run to get a better understanding of the inputs and there results. A few example files are included:

* ROJointExample.toml
* ROSingleExample.toml

Run the Example file versions to see how they can impact the results.

Invoking the model: All the switches for ARetirementPlanner.exe are display by invoking it with the help (–h) switch. The optimizer will print out the following information:



Model input specification:

The basis for the model input specification is the toml specification[[6]](#footnote-6) with its library. The Retirement Optimizer toml grammar[[7]](#footnote-7) and a few comments on semantics is as follows:

Tomlfile := [title] [status] [return] [inflation] [maximize] iam+ SocialSecurity\* income\* expenses\* assets\* [desired | max] IRA\* Roth\* [Aftertax]

Title := ‘title’ ‘=’ string

Status:= ‘retirement\_type’ ‘=’ (‘joint’ | ‘mseparate’ | ‘single’)

Return := ‘returns’ ‘=’ float

Inflation := ‘inflation’ ‘=’ float

Maximize := ‘maximize’ ‘=’ ( ‘Spending’ | ‘PlusEstate’ )

There are a number of objects tied directly to a specific retiree. The iam section is used to define several of them, age, when to retire… The Itext defined in the iam section is then used as a key, label or name for the retiree. Other sections tied directly to the retiree will used the same Itext value. These include sections: IRA, Roth and SocailSecurity. I think of Itext as the iam text. Similarly we define sections with Ltext (here I think Label text) where the Ltext also serves as a label, key or name for the particular item to distinguish the different items of the particular section type. For example, there may be several income sections used each with their own Ltext. One might be a part time job, another rent from a house you own or an annuity. Income, expense and asset sections all use these Ltext labels. While Ltext have no natural limit on how many there can be (you can only have 10 for each section type with the current implementation) Itext are limited to at most two (2) sense they serve as a label or name for a retiree and this application only supports plans for at most two retirees, the retiree and spouse.

Iam := ‘[‘ iam [.Itext] ‘]’ [ primary ] age retire through [ definedPlan ]

Primary := ‘primary’ ‘=’ bool

Age := ‘age’ ‘=’ integer

Retire := ‘retire’ ‘=’ integer

Through := ‘through’ ‘=’ integer

definedPlan := ‘definedContributionPlan’ ‘=’ period

SocialSecurity := ‘[‘ SocialSecurity [.Itext] ‘]’ FRA PIA SSstart

FRA := ‘FRA’ ‘=’ integer

SSstart := ‘age’ ‘=’ openperiod

Income := ‘[‘ income [.Ltext] ‘]’ amount agePeriod inflate tax

Expense := ‘[‘ expense [.Ltext] ‘]’ amount agePeriod inflate tax

Assets := ‘[‘ asset [.Ltext] ‘]’ value costImprove sell owed primaryres rate broker

Value := ‘value’ ‘=’ integer

costImprove := ‘costAndImprovements’ ‘=’ integer

sell := ‘ageToSell’ ‘=’ integer

owed := ‘owedAtAgeToSell’ ‘=’ integer

primaryres := ‘primaryResidence’ ‘=’ bool

Desired := ‘[‘ ‘desired.income’ ‘]’ amount

Max := ‘[‘ ‘max.income’ ‘]’ amount

IRA := ‘[‘ ‘IRA’ [.Itext] ‘]’ balance rate contrib inflate Pperiod

Roth := ‘[‘ ‘Roth’ [.Itext] ‘]’ balance rate contrib inflate Pperiod

Aftertax := ‘[‘ ‘Aftertax’ ‘]’ balance basis rate contrib inflate Pperiod

Basis := ‘basis’ ‘=’ integer

Balance := ‘bal’ ‘=’ integer

rate := ‘rate’ ‘=’ float

contrib := ‘contrib’ ‘=’ integer

Pperiod := ‘period’ ‘=’ period

broker := ‘brokerageRate’ ‘=’ float

Inflate := ‘inflation’ ‘=’ bool

Tax := ‘tax’ ‘=’ bool

agePeriod := ‘age’ ‘=’ period

amount := ‘amount’ ‘=’ integer

period := ‘”’ integer ‘-‘ integer ‘”’

OpenPeriod := ‘”’ integer ‘-‘ ‘”’

Bool := “true” | “false”

Float := integer [ ‘.’ Integer ]

Integer := (0|1|2|3|4|5|6|7|8|9)+

String := ‘”’ text ‘”’

Ltext := text

Itext := text

Text := alphanumeric\*

The basic format is sections of information ‘[‘ section name ‘]’ where section name can be a category followed by a ‘.’ and a descriptive 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.

For those doing a ‘joint’ plan there is one thing that is almost certain to cause confusion and maybe even an error that is difficult to understand and correct without really internalizing the fact that most places where an age or age list is require you use the primary persons age. Exceptions are iam, social security, TDRA and RothRA sections; everywhere else use the primary age line. That is the age for the retiree with ‘primary = true’ in their iam section. A perfect example of how this can trip us up is if we define something like [income.retiree2] using retiree2’s age line rather than the primary age line.

Let us step through the ‘StartHere’ toml file to understand the input file components.

General model information (global section):

# This input file is meant as a starter for a joint plan

# most fields are options and have reasonable defaults

# defaults to joint, could also be single or

# mseparate(married filing separately)

retirement\_type = 'joint'

returns = 6 # return rate of investments, defaults to 6%

inflation = 2.5 # yearly inflation rate, defaults to 2.5%

# optimize for what?

# - 'Spending' or

# - spending 'PlusEstate',

# defaults to ‘Spending’

maximize = "PlusEstate"

Retiree Section Retiree is specified. If only one retiree is given the final portion of iam.final can be dropped. However it must match with the final portion of TDRA and RothRA accounts. That is if this final portion is dropped here it must also be dropped for the accounts. ([iam] mates with [IRA] and [iam.joe] mates with [IRA.joe]). If you or your spouse is working and qualify for a defined contribution plan like a 401(k), 403(b) or TSP type TDRA you can use the definedContributionPlan specifier to indicate the years you may contribute. This has the effect in the model of increasing your maximum allowable contribution to take into account their contribution allowances:

# iam (for each) is required in some joint cases (".xxx" is

# used to match accounts IRA/roth)

[iam.retiree1]

primary = true # retiree listed first in the output (must choose one)

age = 54 # your current age

retire = 65 # age you plan to retire

through = 75 # age you want to plan through

definedContributionPlan = "54-64" # Years for a 401(k), 403(b), TSP

A second retiree means that at least one must have the final portion of iam.final specified. It is possible to only specify one retiree for a joint plan if all tax deferred accounts are owned by the specified retiree.

# iam (for each) is required in some joint cases (".xxx" used

# to match accounts IRA/roth)

[iam.retiree2]

# primary = false # primary left out defaults to false

age = 54 # your current age

retire = 65 # age you plan to retire

through = 75 # age you want to plan through

#definedContributionPlan = "54-64" # Years for a 401(k), 403(b), TSP

The social security section is optional. If given its final portion as in SocialSecurity.final must match the final portions given or not in the retiree section.

# Social Security section assumes inflation and 85% taxed

[SocialSecurity.retiree1]

FRA = 67 # your full retirement age (FRA) according to the IRS

amount = 20\_000 # estimated yearly amount at FRA;

age = "70-" # period you expect to receive SS

# ("68-" indicates start at 68 and continue)

[SocialSecurity.retiree2]

FRA = 67 # your full retirement age (FRA) according to the IRS

amount = -1 # -1 or 0 for default spousal benefit amount

age = "67-" # period you expect to receive SS ("68-"

# indicates start at 68 and continue)

Annuities, reverse mortgages and other types of income can be specified in an income section. These sections are named as income.final where the final portion is some name meaningful to you for the type of income it is. It includes an annual amount, period you will receive it, whether it should be inflation adjusted and whether it is taxable.

[income.taxfreeNoneInflationAdjustedAnuity]

amount = 3000 # yearly amount

age = "65-70" # period you expect to receive it

inflation = false # not inflation adjusted (default false)

tax = false # not federally taxable (default false)

[income.InflationAdjustedAnuity]

amount = 3000 # yearly amount

age = "65-70" # period you expect to receive it

inflation = true # inflation adjusted (default false)

tax = true # federally taxable (default false)

Reverse mortgages take many forms with one of the simplest paying a constant amount each month for the rest of your life. This form and others can be represented in the income section. For example the simple form can be represented as follows:

[income.reversemortgage]

amount = 12000 # yearly amount, 1000/mo

age = '70-' # period to receive payments

inflation = false # payment is not inflation adjusted

#(default false)

tax = false # payment/loan is not taxable (default false)

[income.rental]

amount = 5\_000 # yearly amount

age = "67-" # period you expect to receive it

inflation = true # inflation adjusted (default false)

tax = true # federally taxable (default false)

Major assets like your home or other real estate can have a significant impact on your retirement. If you plan to sell these assets during your retirement you can include there return in your plan with the [asset] section as below. This section provides the information needed for the optimizer to determine the amount to add to your after tax account and the amount that is taxable. If the asset is your primary residence the IRS exempts the first $250K ($500K joint) of capital gains from taxes.[[8]](#footnote-8)

[asset.home]

value = 550\_000 # current value of the asset

costAndImprovements = 300\_000 # purchase price plus improvement cost

ageToSell = 72 # age at which to sell the asset

owedAtAgeToSell = 100\_000 # owed at time of sell (ageToSell)

primaryResidence = true # Primary residence gets tax break

rate = 4 # asset rate of return

# (defaults to global rate)

brokerageRate = 4 # brokerage fee percent (default 4%)

[asset.rental]

value = 250\_000 # current value of the asset

costAndImprovements = 150\_000 # purchase price plus improvement cost

ageToSell = 72 # age at which to sell the asset

owedAtAgeToSell = 100\_000 # owed at time of sell (ageToSell)

primaryResidence = false # Primary residence gets tax break

rate = 4 # asset rate of return

# (defaults to global rate)

brokerageRate = 4 # brokerage fee percent (default 4%)

Desired income and max income should only be used with the appropriate ‘maximize’ case

[desired.income] # used when maximize = "PlusEstate"

amount = 45\_000 # retirement first year income

[max.income] # used when maximize = "Spendable" (default)

amount = 100\_000 # retirement first year income

Account sections for TDRA, RothRA and ATRSI all contain a bal and rate options for the account balance and account specific rate or return. If the rate is not given the global rate will be used. The final portion of the account.final section name must match with the final portion of the retiree section for the account owner. ATRSI accounts have an additional field ‘basis’ for the amount of money contributed to the account. This is after tax money and will not be taxed on withdrawal.

# pre-tax IRA accounts (TDRA)

[IRA.retiree1]

bal = 200\_000 # current balance

rate = 7.25 # defaults to global rate set above

contrib = 0 # Annual contribution to make for period (below)

inflation = false # Will the contribution rise with inflation?

# (default false)

period = '56-60' # period you will be making the contributions

[IRA.retiree2]

bal = 100\_000 # current balance

rate = 7.25 # defaults to global rate set above

contrib = 0 # Annual contribution to make for period below

inflation = false # Inflate contribution with inflation rate?

# (default false)

period = '56-60' # period you will be making the contributions

# roth IRA accounts (RothRA)

[roth.retiree1]

bal = 20\_000 # current balance

rate = 7.25 # defaults to global rate set above

contrib = 0 # Annual contribution to make for period (below)

inflation = false # Inflate contribution with inflation?

# (default false)

period = '56-60' # period you will be making the contributions

[roth.retiree2]

bal = 20\_000 # current balance

rate = 7.25 # defaults to global rate set above

contrib = 0 # Annual contribution to make for period (below)

inflation = false # Will the contribution rise with inflation?

# (default false)

period = '56-60' # period you will be making the contributions

# after tax savings accounts (ATRSI)

[aftertax]

bal = 100\_000 # current balance

basis = 50\_000 # Contributions to total, for capital gains tax

rate = 7.25 # defaults to global rate set above

contrib = 0 # Anual contribution to make for period (below)

inflation = false # Will the contribution rise with inflation?

# (default false)

period = '56-60' # period you will be making the contributions

1. This document is for the Golang version of the Retirement Planner. Earlier versions where for the python prototype. [↑](#footnote-ref-1)
2. 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-2)
3. 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-3)
4. 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-4)
5. 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-5)
6. The toml specification is archived at: <https://github.com/toml-lang/toml> [↑](#footnote-ref-6)
7. We use a BNF expression for the grammar. Items inclosed in ‘[‘ and ‘]’ are optional, items followed by a ‘+’ must appear at least once possible more, items followed by a ‘\*’ can appear zero or more times. The ‘|’ is a logical or, ‘(‘ and ‘)’ is simple grouping and items in single quotes are literals and much be used as shown. [↑](#footnote-ref-7)
8. Capital Gains exclusions for your primary residence remain the same in 2018 with the new tax plan as they were in 2017. [↑](#footnote-ref-8)