BANK OF CANADA
BANQUE DU CANADA

Canadian Debt Strategy Model User Guide

Purpose and Structure

The overall purpose of the Canadian Debt Strategy Model (CDSM) User Guide is to provide a general user with the information required to operate the Model by focusing on the context and key elements.

The CDSM is entirely written in the software application Matlab and run in a Linux server[1]. There are three distinct steps or sub-models of the CDSM. The first step generates macroeconomic and interest rate scenarios (***generateScenarios_new.m***). The second step creates representative debt strategies (***makeStrategies.m***) and evaluates the cost and risk of those strategies under those scenarios (***main.m***). The third step produces a frontier of cost/risk-efficient strategies based on those results (***optim_main.m***).

Section 1 of this guide provides background information on the CDSM. Sections 2, 3, and 4 provide a process flow chart and operating instructions for each of the three sub-models. Section 5 provides a summary of how the CDSM results are applied.

An interactive guide to the complete coding of the CDSM can be found in the **documentation** folder.

1 Background

1.1 Canada's debt management objectives[2]

The fundamental objectives of debt management are to raise stable and low-cost funding to meet the financial needs of the Government of Canada and to maintain a well-functioning Government of Canada securities market. Achieving stable, low-cost funding involves striking a balance between the cost and risk associated with the debt structure as funding needs

[1] The model with minor syntax adjustments can also be run in Windows, but with a possibly less feasible computing time. It has also been run on an HPC cluster.

[2] Budget 2017 – Budget Plan: Annex 2 – Debt Management Strategy for 2017-18

change and under various market conditions. Having access to a well-functioning securities market ensures that funding can be raised efficiently over time to meet the Government's needs. Moreover, to support a liquid and well-functioning Government of Canada securities market, the Government strives to promote transparency and regularity.

1.2 The development and role of the CDSM

The CDSM is only one of several tools available to Canadian debt managers to assist in the debt management policy recommendations. The CDSM was developed at the Bank of Canada with support from the Department of Finance. It is a steady-state, stochastic simulation model that integrates macroeconomic, interest rate, and fiscal conditions. It also has the flexibility to incorporate multiple issuance and maturity structures and to consider different risk measures in optimization. As with all models, the CDSM requires some simplifying assumptions.[3]

The CDSM is used by debt managers to compare different strategies for the financing of the government's debt. It also provides insights into the complex interactions involved and the impact of different variables on outcomes. With knowledge of the specific topical questions in debt management, the user can synthesize the large amount of information the model provides into relevant points for policy.

1.3 How a debt manager uses the model

A government has a set of possible financing instruments (i.e. treasury bills, nominal coupon bonds, or inflation-linked bonds in Canada's case[4]) that it can use to refinance maturing debt and/or budgetary deficits. The debt manager can choose the relative mix of these financing instruments as well as the maturity composition (i.e. 2-year, 3-year, 5-year, 10-year, or 30-year nominal bonds) within each instrument type. This collection of choices is defined as the government's financing strategy and is the **choice variable** that the model is solving for.

As models are meant to be a simplification of reality, it is up to the debt managers using this model to specify it in a way that adequately reflects their reality – so that its results are useful for decision-making. These specifications form the **model parameters**, and include things like the interest rate model used or the issuance structure for a given instrument (e.g. how frequently a new 2-year bond is issued). For illustrative purposes, the published code and procedures are based on just one possible set of model parameters.[5]

In addition, the debt manager must ensure that the model inputs reflect their existing policy considerations. Thus, certain parts of the code require their own **policy input**. One example is the minimum issuance constraints, which are set such that the decision space covers only what the debt manager views as feasible financing strategies. Those parts of the code are

[3] One example of this is the absence of buybacks in the model. For details on technical aspects of the model, see the code documentation or Additional Reading.

[4] For simplicity reasons, this excludes retail and other marketable debt.

[5] They do not represent the official parameters used by Canada when running the CDSM.

left blank here (with ---) and will need to be filled in for the model to run. These inputs are specific to each debt manager, and are reviewed and updated as part of the regular operation of the model.

1.4 Stochastic simulation of steady-state

The Canadian Debt Strategy Model is meant to be run as a steady-state model.[6] This means that everything is based on a hypothetical permanent future where interest rates, macroeconomic variables, and debt stocks vary around a constant average, i.e. are stationary. This model uses stochastic Monte Carlo simulation to derive, through repeated experiments, the properties of this hypothetical steady-state environment.

In this steady-state environment, it follows that the financing strategy choice variable must also be stationary**. The financing strategy is thus defined by a single vector, representing a constant target allocation between instruments in steady-state.** The different financing strategies are evaluated by computing their summary statistics under these steady-state stochastic simulations.

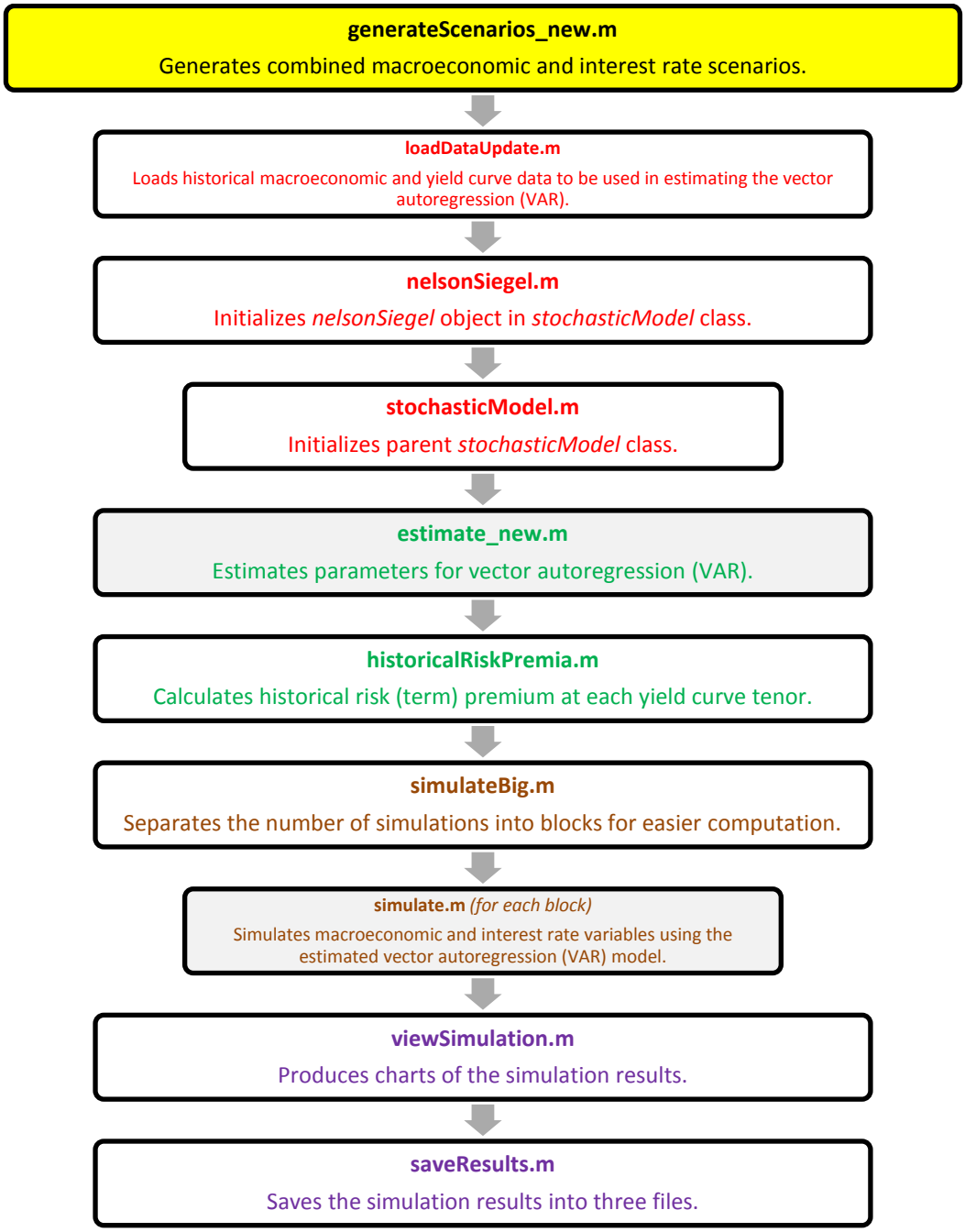
2 Generating Scenarios

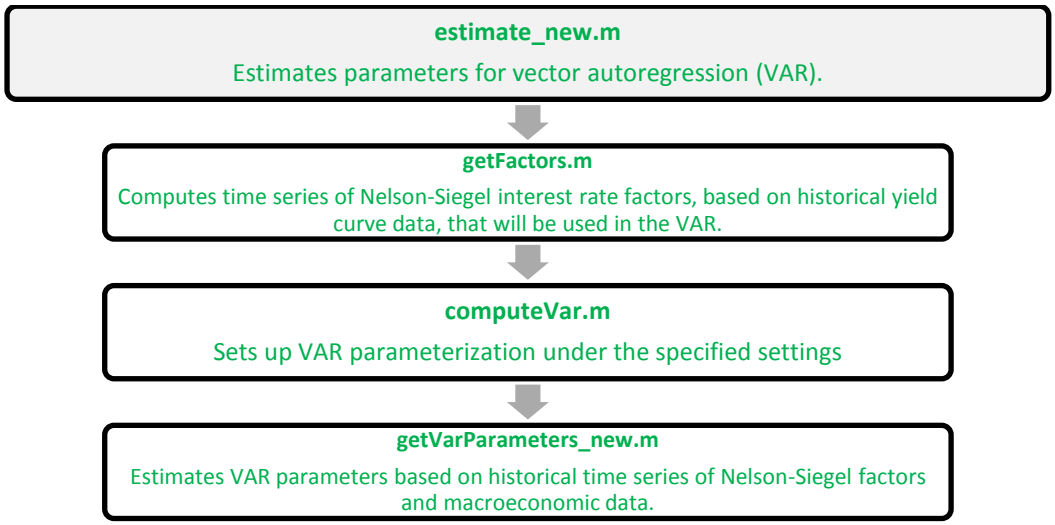
The first step of the model is to simulate these steady-state interest rate and economic scenarios under which the different financing strategies are evaluated. For this, several macroeconomic term-structure models have been tested and are implementable in the CDSM. The Nelson-Siegel model described here provides the "base case" results.

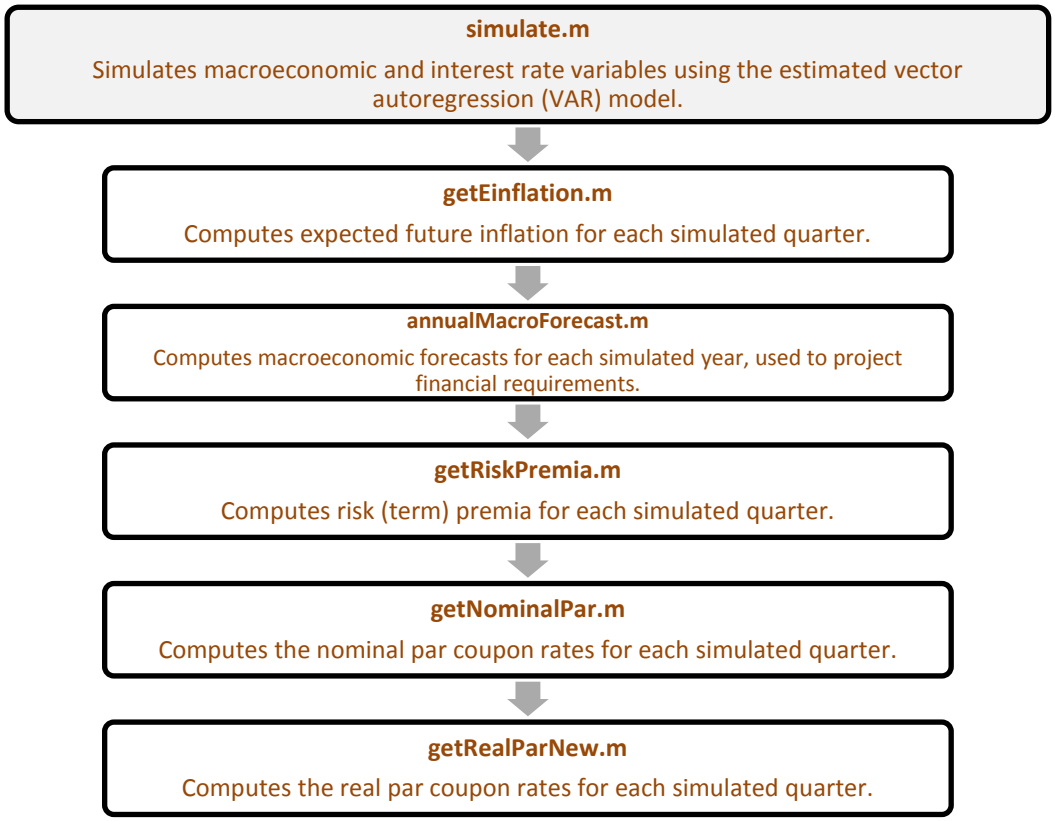
This step is done in the **stochastic** folder through the ***generateScenarios_new.m*** program, which has multiple nested sub-programs. The flow charts below summarize ***generateScenarios_new.m*** as well as its two main sub-programs: ***estimate_new.m*** and ***simulate.m***.

[6] Although the code structure is flexible for allowing non-steady-state model parameters (e.g. an increasing interest rate), those require a fundamentally different interpretation of the results and are not typically used.

2.1 Program: generateScenarios_new.m




estimate_new.m
Estimates parameters for vector autoregression (VAR).
getFactors.m
Computes time series of Nelson-Siegel interest rate factors, based on historical yield curve data, that will be used in the VAR.
computeVar.m
Sets up VAR parameterization under the specified settings
getVarParameters_new.m
Estimates VAR parameters based on historical time series of Nelson-Siegel factors and macroeconomic data.


simulate.m
Simulates macroeconomic and interest rate variables using the estimated vector autoregression (VAR) model.
getEinflation.m
Computes expected future inflation for each simulated quarter.
annualMacroForecast.m
Computes macroeconomic forecasts for each simulated year, used to project financial requirements.
getRiskPremia.m
Computes risk (term) premia for each simulated quarter.
getNominalPar.m
Computes the nominal par coupon rates for each simulated quarter.
getRealParNew.m
Computes the real par coupon rates for each simulated quarter.

2.2 Procedures

1. In the six macroeconomic data files: ***ovnrateMMMYY.txt***, ***inflyearMMMYY.txt***, ***inflcoreyearMMMYY.txt***, ***ygapMMMYY.txt***, ***potgdppctchquarterMMMYY.txt***, and ***RGDPpctchquarterMMMYY.txt*** in the **varData** subfolder, update to the most recent month.
    - The six files are monthly historical time series for each macroeconomic variable: (1) overnight rate, (2) inflation, (3) core inflation, (4) output gap, (5) potential GDP growth, and (6) real GDP growth. The choice of data source is a policy input.
2. In the ***zeroCurvesMMMYY.txt*** interest rate data file in the **varData** subfolder, update to the most recent month.
    - The file is a monthly historical time series for each yield curve point. The choice of data source is a policy input.
3. In ***loadDataUpdate.m***, update the data file names **(Line 45, 52, 59, 66, 77, 93, 115)** and relevant historical period **(Lines 33-34)** accordingly.
    - The relevant historical period is a policy input for the historical period of data that is being used to parameterize the simulation model. It is meant to be representative of steady-state.
4. In ***generateScenarios_new.m***, set the number of scenarios **(Line 18)** as well as long-term means for each macro variable **(Lines 35-39)** and Nelson-Siegel factor **(Lines 44-46)**.
    - Long-term mean is a policy input for the assumed average of the variable in steady-state.
5. In ***generateScenarios_new.m***, update the path and base name of the saved simulation results files accordingly **(Line 121)**.
6. Run ***generateScenarios_new.m*** on MATLAB in Linux.
    - This will create three files with the results of the simulation.

3 Creating a Training Set

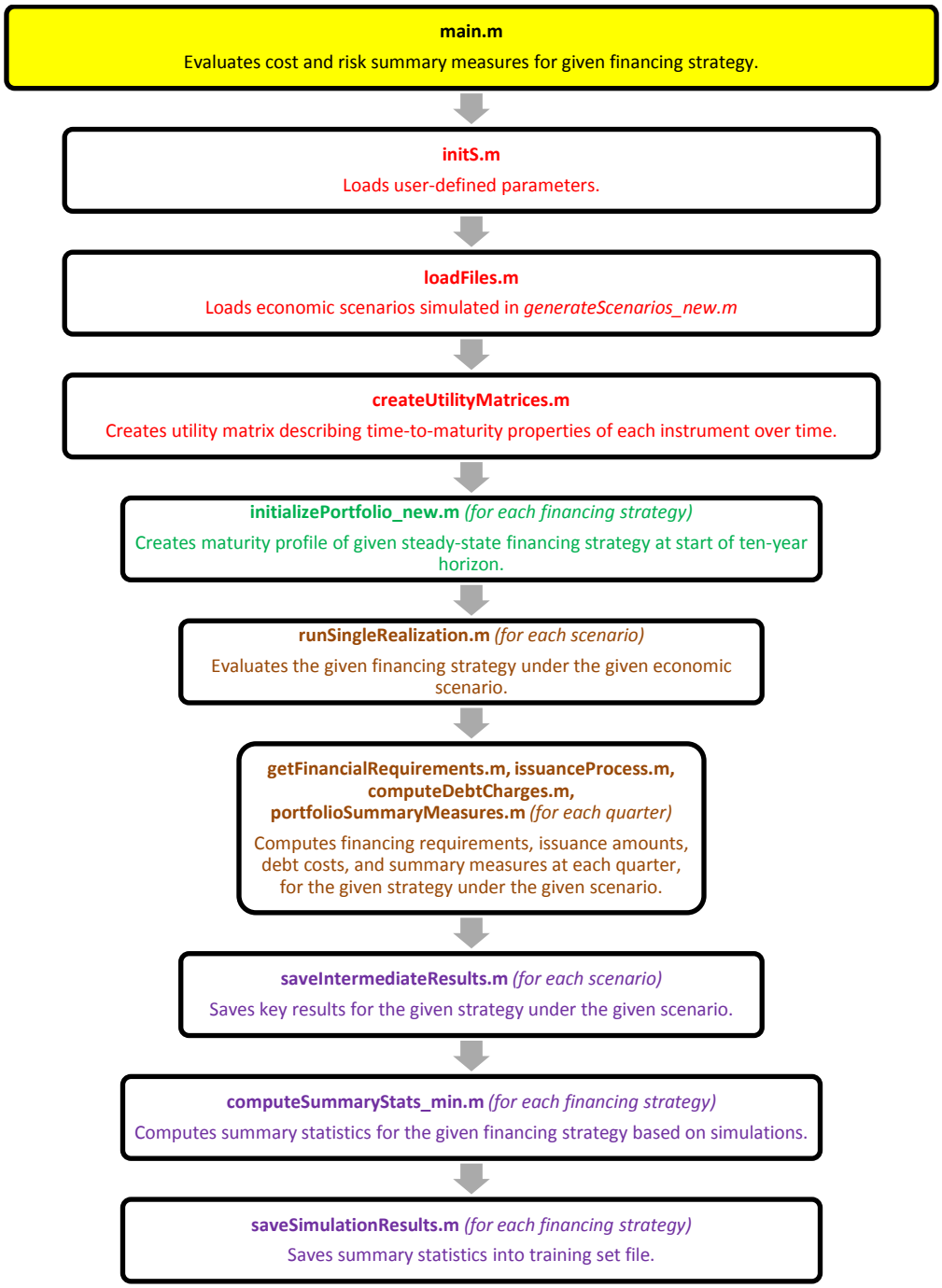
The second step of the model is to create a representative set of financing strategies and evaluate them under the simulated economic scenarios from the first step. The evaluated cost and risk summary measures for these strategies form a training set, through which a general function of cost and risk for any strategy can be curve-fitted in the third step.

Creating a set of strategies is done in the **model** folder through the ***makeStrategies.m*** program. The strategies are defined such that they cover the full space of possible financing strategies with an emphasis on feasible strategies, for accurate and efficient curve-fitting.

Evaluating each strategy is done in the **model** folder through the ***main.m*** program, which has multiple nested sub-programs. The evaluation is done scenario-by-scenario across a ten-year horizon on a quarterly basis. This model uses a Linux shell script, ***runAllProduc-***

tion.csh, to run ***main.m*** in parallel loops across multiple blocks of strategies at the same time. The flow chart below summarizes ***main.m***.

3.1 Program: main.m



3.2 Procedures

3.2.1 Making Strategies

1. In ***makeStrategies.m***, set the number of representative strategies to evaluate and the size of each parallel block to evaluate them with **(Line 30, 33)**.
    - The more strategies are evaluated, the more consistent the results will be. The number of blocks chosen will depend on the server capacity and time-sensitivity of the results.
2. In ***makeStrategies.m***, set the vector of minimum constraints **(Line 57)**.
    - This vector is a policy input for the minimum amount that can be allocated, on a percentage basis, to each instrument. At this step, it informs the representative set of strategies being evaluated for the training set.
3. In ***makeStrategies.m***, update the name of the saved strategy files accordingly **(Line 114)**.
4. Run ***makeStrategies.m*** on MATLAB in Linux.
    - This will create blocks containing data on all the representative strategies.

3.2.2 Setting up the Debt Charge Engine

5. In ***main.m***, ensure that it is evaluating a strategy from the *strategyFile* variable and not a pre-defined strategy vector **(Lines 72-73)**.[7]
6. In ***initS.m***, set the appropriate number of strategies in each block as well as the number of scenarios, out of the full set of generated scenarios, under which each strategy will be evaluated **(Line 25, 27)**.
7. In ***initS.m***, update the steady-state debt stock value **(Line 52)**.
    - This value is a policy input that is based on an internal assumption of what debt stock would be in this hypothetical steady-state. It can be estimated from fiscal projections and/or other internal sources.
8. In ***initS.m***, update the values for initial GDP and initial government revenues **(Line 53, 79)**.
    - These values are policy inputs based on internal assumptions of GDP and revenues in steady-state.
9. In ***initS.m***, update the coefficients on each macroeconomic variable in the equation for government expenses and revenues growth **(Lines 75-76)**.
    - These values are policy inputs based on internal fiscal assumptions and/or parameterization. See Robbins, Torgunrud, and Matier (2007) for details.

[7] Separate from this procedure, the user can evaluate pre-defined individual strategies instead by running ***main.m*** in MATLAB in Linux directly. Note the other setting in **saveSimulationResults.m**, **Lines 24-25**.

10. In ***initSpenF_wts.m***, update the parameters of the penalty functions.
    - These are policy inputs defining the range of reasonable issuance amounts for each instrument, and the additional costs – assumed and/or parameterized – that would be incurred for issuance amounts outside of that range. Ignore this if the penalty function setting, a policy input itself, is turned off (***initS.m*, Line 56**).
11. In ***loadFiles.m***, update the names of the three loaded data files to match those of the generated scenario files **(Lines 21-23)**.
12. In ***saveSimulationResults.m***, ensure that it is saving under the *resultsFile* variable name and not under a pre-specified name **(Lines 29-30)**.

3.2.3 Running the Debt Charge Engine on the Server

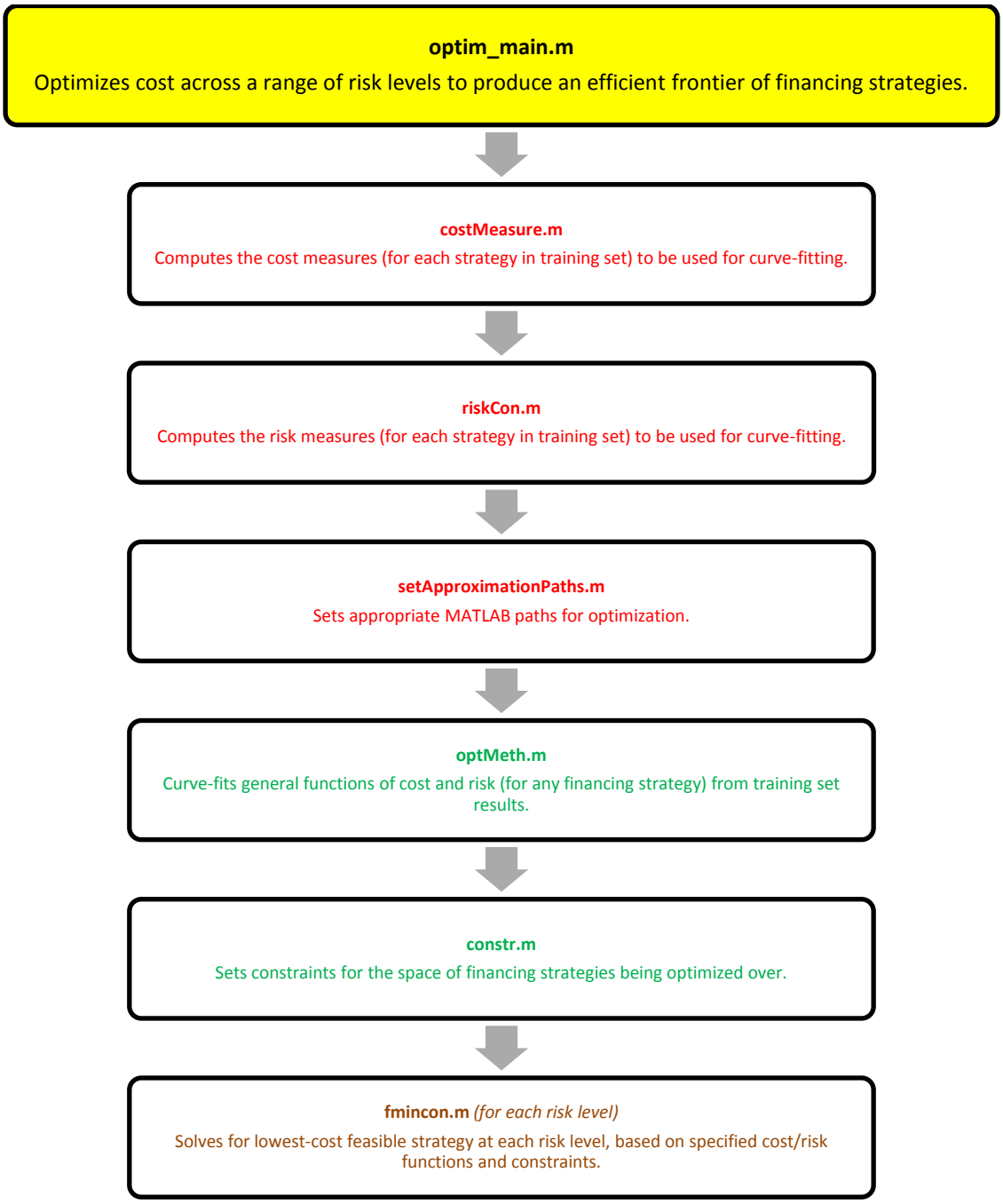
13. In the strategy block list file ***portFilesServer_MMMYY.txt*** in the **results** folder, update to match the names of strategy blocks created by ***makeStrategies.m***.
    - The Linux shell script calls this strategy list file to run parallel loops of ***main.m*** over each of the listed strategy blocks.
14. In the ***runAllProduction.csh*** Linux shell script in the **model** folder, update the names of the strategy block list file **(Line 6)** and log file **(Line 4)** accordingly.
15. In the Linux terminal in the **model** directory, run the Linux shell script with the command: "***nohup csh <runAllProduction.csh> out_MMMYY.txt***"
    - This will start the process for evaluating the strategies. The whole thing can take from an hour to a few days depending on the settings and server. Once completed, files with the full results for each block will be saved in the **results/ns/trainingSets/tsBlocks/** folder.
16. In ***mergeResults.m*** in the **model** folder, set the stopping point for the *while* loop to 1 plus the number of blocks **(Line 21)**.
17. Run ***mergeResults.m*** in MATLAB in Linux.
    - This will combine the result files for each block into a single file. This is the training set file.
18. Move the training set file from the **results/ns/trainingSets/tsBlocks/** subfolder to the **results/ns/trainingSets/** subfolder.

4 Producing an Efficient Frontier

The third and final step of this model is to use the training set results from the second step to curve-fit a general function of cost and risk for any strategy, and use that function to optimize for the lowest-cost strategy at multiple fixed risk levels. The final result is an efficient frontier describing the efficient financing strategies available to the debt manager, and the risk-cost trade-off between them.

This step is done in the **optimize** folder through the ***optim_main.m*** program, which has multiple nested sub-programs. The flow chart below summarizes ***optim_main.m***.

4.1 Program: optim_main.m



4.2 Procedures

1. In ***optim_main***, update the vector of minimum constraints **(Line 29)**.
    - This vector is the same policy input as was used in creating strategies, i.e. the minimum percentage amount that can be allocated to each instrument. At this step, it restricts the space of strategies we are optimizing over.
2. In ***optim_main***, define the risk measure **(Line 35)**.
    - This string is a policy input based on what the debt manager views as the relevant risk measure. Common risk measures computed in the model, and their matching strings, can be found in ***riskCon.m***.
3. In ***optim_main***, define the relevant range of efficient frontier points; i.e. set the starting, ending, and interval risk values at which the cost-minimizing optimization will be done **(Lines 74-76)**.
    - This requires some trial and error. In the end, it needs a set of points that: (1) all have a positive *e_flag* value (i.e. optimization result is usable), and (2) leads to an informative result for decision-making (i.e. covers a sufficient range with sufficient granularity). The specific criteria for (2) is a policy input. This will become easier as the user gets more familiar with the range of relevant values.
4. Run ***optim_main*** on MATLAB in Linux.
    - This will save a file with the optimization results.
5. Using these optimization results, construct an efficient frontier using the ***par***, ***fstar***, and ***xstar*** variables.
    - ***par*** gives the set of risk value points (x-axis) via the starting, ending, and interval values
    - for each risk value point, ***xstar*** gives the lowest-cost financing strategy with that risk value or lower
    - for each risk value point, ***fstar*** gives that lowest cost for that strategy (y-axis)

5 Applying Model Results

From this final efficient frontier, the debt manager can get a sense of the cost-risk trade-offs and the associated efficient steady-state financing strategies. It is now up to the debt manager themselves to decide how they wish to use this information, with an understanding of the specific model parameters, assumptions, and policy inputs.

Canada's debt managers use an internal version of the CDSM as one of the tools in their decision-making process, both on a standardized basis to inform the Government of Canada's annual Debt Management Strategy, and on an ad hoc basis to provide context for various policy decisions. This is done with a regular review and update of the model parameters and

policy inputs being used. It is also done with an understanding of the limitations that a model like this has in capturing the full scope of the debt management problem.

More specifically, as a steady-state model, the model results are by nature hypothetical. In addition, as a simplification of the real world, the model cannot reflect the full range macroeconomic, fiscal, market, and other factors as they relate to the Government's objectives.

The CDSM therefore only plays a partial role in Canada's ultimate debt strategy decisions. As with any sovereign, debt management in Canada is a holistic process that requires considerable combination of analysis, modelling, discussion, consultation, and judgment on the part of debt managers and other policymakers.

Appendix: Additional Reading

Bolder, D.J. 2001. “Affine Term-Structure Models: Theory and Implementation.” Bank of Canada Working Paper No. 2001-15.

———. 2003. “A Stochastic Simulation Framework for the Government of Canada’s Debt Strategy.” Bank of Canada Working Paper No. 2003-10.

———. 2006. “Modelling Term-Structure Dynamics for Risk Management: A Practitioner’s Perspective.” Bank of Canada Working Paper No. 2006-48.

———. 2008. “The Canadian Debt-Strategy Model.” *Bank of Canada Review* (Summer): 3-16.

Bolder, D.J. and S. Deeley. 2011. “The Canadian Debt Strategy Model: An Overview of the Principal Elements.” Bank of Canada Staff Discussion Paper No. 2011-3.

Bolder, D.J. and S. Gusba. 2002. “Exponentials, Polynomials, and Fourier Series: More Yield Curve Modelling at the Bank of Canada.” Bank of Canada Working Paper No. 2002-29.

Bolder, D.J. and S. Liu. 2007. “Examining Simple Joint Macroeconomic and Term-Structure Models: A Practitioner’s Perspective.” Bank of Canada Working Paper No. 2007-49.

Bolder, D.J. and T. Rubin. 2007. “Optimization in a Simulation Setting: Use of Function Approximation in Debt Strategy Analysis.” Bank of Canada Working Paper No. 2007-13.

Robbins, J., B. Torgunrud, and C. Matier. 2007. “Fiscal Planning Under Uncertainty: The Implications of Economic and Fiscal Uncertainty for Budget Forecasts.” Presented at Banca d’Italia Workshop in Perugia, March 29–31, 2007