

MANITOBA TYPE 2 DIABETES MELLITUS INDIVIDUAL OPTIMIZATION MODEL—USER GUIDE

The intent of this document is to furnish prospective users of the Manitoba Type 2 Diabetes Mellitus (T2DM) Individual Optimization Model (hereafter, the “decision model”) with the background information necessary to set up, solve, and apply a dynamic stochastic optimization problem concerning individual health-related decision-making among those at risk for T2DM in the Province of Manitoba. The document provides step-by-step instructions on how to successfully deploy the model, prepare and solve the optimization problem, implement sensitivity analyses around key parameters, and design and execute simple microsimulation exercises using model-generated results.

Prior to, or concurrent with the review of the document, the reader is strongly encouraged to consult the thesis motivating and supplying both the conceptual foundation for the decision model and a complete list of references for all data sources used to select parameter values. It furthermore explains how to interpret model-generated outputs, including those produced by the supplementary microsimulation module. The thesis is freely available to the public at the following location on MSpace, an online repository for intellectual output created by researchers associated with the University of Manitoba (2018):

<LINK TO ZIP FILE HERE>

Prospective users may also benefit from consulting an online demonstration developed by the author, which covers roughly the same subject matter presented below. The demonstration is accessible at: https://youtu.be/1u50_bkacx0.

A.1 OVERVIEW

The decision model consists of an integrated collection of programs developed by the author in R, MATLAB and Excel Visual Basic for Applications. The model comes packaged with or is programmed to acquire all materials required to replicate the findings presented in the thesis, excepting those listed below. It processes and compiles these materials into a format appropriate for integration into the dynamic optimization problem, which is then solved by applying the backward induction algorithm. The steps involved in implementing the decision model are depicted in Figure 1 below.

The model outputs the decision-maker's optimal policy and value function, which may themselves be of interest but can also be used to drive microsimulation exercises that illustrate how the projected prevalence and incidence of T2DM in Manitoba, and direct and/or indirect losses attributable to the condition, may vary in populations with prespecified characteristics in different user-defined scenarios.

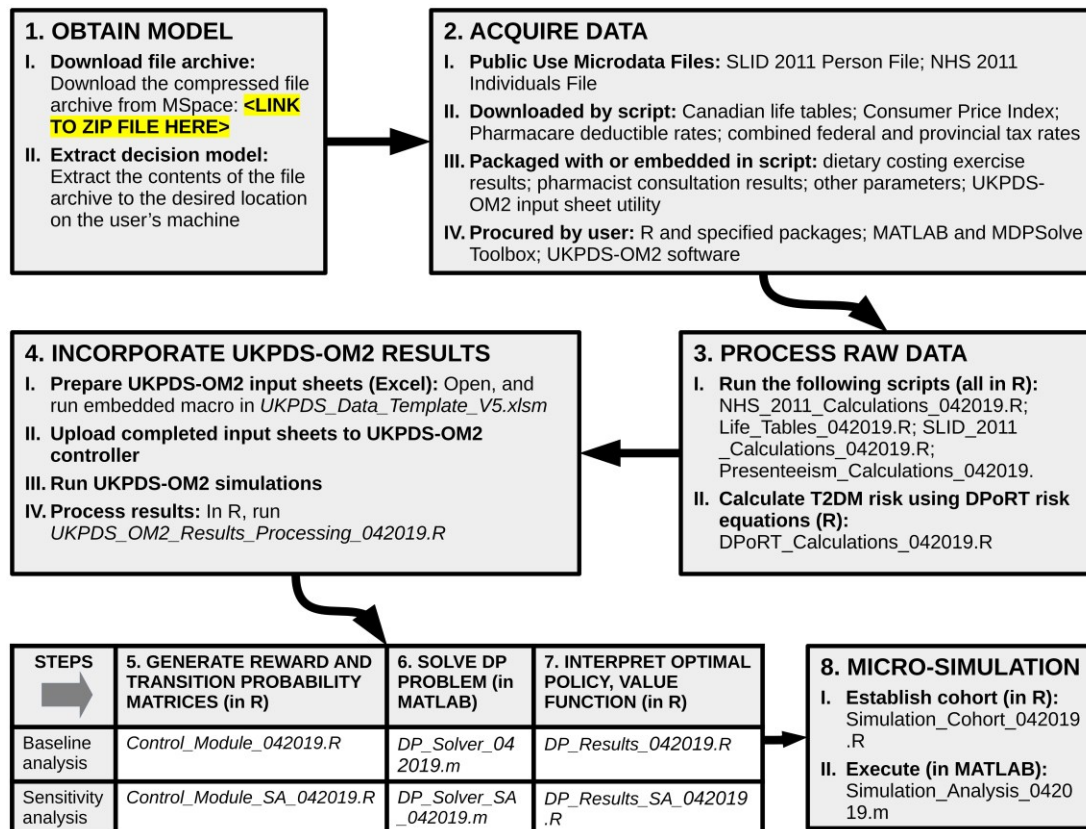


Figure 1: “Cheat sheet” for the implementation and application of the decision model

The remainder of this document walks the user through the implementation and solution of the decision model, with reference to the steps depicted in Figure 1.

A.2 STEP 1—OBTAINING THE MODEL

Once downloaded from MSpace, the contents of the file archive can be extracted to any location on the user’s machine (the reference to the C:\ directory in Table 1 below is purely illustrative). The user should not attempt to manipulate the model’s file structure, as this may detrimentally impact its operation; however, they should be familiar with this structure to facilitate access to model outputs and, where necessary, to permit troubleshooting:

Table 1: Decision model file structure	
Directory	Description
C:\...\Diabetes_Model_042019	This is the base directory, containing the R and MATLAB scripts required to assign parameter values for and solve the decision model, and to interpret and apply the results.
C:\...\Diabetes_Model_042019\MATLAB	This folder contains the reward and transition probability matrices driving the solution of the DP problem, as well as the resulting optimal policies and value functions. It also contains the results of the microsimulation exercise described in Section 5 in Chapter 4 of the thesis.
C:\...\Diabetes_Model_042019\R_Objects	This folder contains intermediate calculations performed using a combination of data from a variety of sources, the results of which will be utilized in downstream calculations.
C:\...\Diabetes_Model_042019\Raw_Data	This folder holds data sources feeding into the process of assigning values or ranges to model parameters. As the directory name suggests, the contents generally should not be renamed or manipulated outside the program itself.
C:\...\Diabetes_Model_042019\UKPDS_OM2	This folder holds all the materials associated with the UKPDS-OM2, including the utility developed to facilitate preparation of the input sheets, the input sheet template, and the UKPDS-OM2 simulation outputs. It is recommended that a shortcut to the UKPDS-OM2 controller also be stored here, although this is not required.

A.3 STEP 2—ACQUIRING THE DATA

Two types of inputs are required to operate the decision model. First, as implied by Figure 1, the user must possess a working copy of R, RStudio, MATLAB and Excel.¹ They must also acquire and install the R packages listed in Table 2, which extend the functionality of the R software environment; the MDPSolve Toolbox, freely available from a website maintained by Paul Fackler (2014), which includes a function that implements the backward induction algorithm in MATLAB; and the United Kingdom Prospective Diabetes Study Outcomes Model v2 (UKPDS-

¹ The model has been thoroughly tested and functions as intended using R version 3.5.3 (via version 1.2.1335 of the RStudio integrated development environment for R), MATLAB 2018a, and Excel 2016. Versions of all R packages I employed in running the model are listed in Table 2. As with any piece of software, compatibility with programs other than those listed there (e.g., substituting Octave for MATLAB) or different versions of these programs is difficult to predict in advance.

OM2), which is licensed free of charge to academic organizations by the University of Oxford (Diabetes Trials Unit, 2018).

Table 2: R packages employed in the decision model		
❖ CANSIM2R (1.14.1)	❖ httr (1.4.0)	❖ readxl (1.3.1)
❖ data.table (1.12.2)	❖ jsonlite (1.6)	❖ reshape2 (1.4.3)
❖ dplyr (0.8.0.1)	❖ knitr (1.22)	❖ rvest (0.3.3)
❖ formattable (0.2.0.1)	❖ lubridate (1.7.4)	❖ stringr (1.4.0)
❖ ggplot2 (3.1.1)	❖ magrittr (1.5)	❖ tibble (2.1.1)
❖ gsubfn (0.7)	❖ openxlsx (4.1.0)	❖ tidyr (0.8.3)
❖ here (0.1)	❖ pdftools (2.2)	❖ xlsx (0.6.1)
❖ htmlTable (1.13.1)	❖ purrr (0.3.2)	❖ xtable (1.8-3)

Important: Before proceeding, the user must set the default working directory in RStudio to the base directory defined in Table 1 above, as this enables the model to identify the location from which scripts are being run. This can be accomplished by accessing the Global Options interface through the Tools menu and selecting the appropriate directory under “R Sessions” in General options.

Second, the decision model requires data from several sources to support the assignment of parameter values. This information falls into the categories listed in Table 3. The model either includes, or is programmed to automatically acquire all necessary inputs except the 2011 National Household Survey (NHS) and 2011 Survey of Labour and Income Dynamics (SLID) Use Microdata Files (PUMFs), which the user must obtain and store in the location specified below:

Table 3: Overview of model inputs		
Input type	Description	User considerations
Microdata files	2011 NHS and 2011 SLID PUMFs are used to calculate household income and to	The PUMFs are freely available to Canadian post-secondary educational institutions participating in the Data Liberation Initiative.

Table 3: Overview of model inputs		
Input type	Description	User considerations
	derive employment state transition probabilities.	Once acquired, these files should be stored in <i>C:\...\Diabetes_Model_042019\Raw_Data\</i> . The files must not be renamed, nor should they be manipulated in any way.
Information downloaded by the model	This includes life tables for Canada and the Consumer Price Index available from the Statistics Canada website (Statistics Canada, 2018), and combined federal and provincial personal income tax rates for 2018 (EY, 2018).	This information is collected automatically by the program through the application of web-scraping techniques. The user must therefore possess an active Internet connection when the associated scripts (<i>NHS_2011_Calculations_042019.R</i> and <i>Life_Tables_042019.R</i>) are run for the model to function as intended.
Information packaged with the model	A few data sources are packaged with the model on MSpace. These include summary tables from the pharmacist consultation and the dietary costing exercise.	This information is stored in <i>C:\...\Diabetes_Model_042019\Raw_Data\</i> . Users not well-acquainted with the model or with relatively little programming experience are advised to refrain from renaming or otherwise manipulating these files.
Other inputs	The values of all remaining parameters are embedded directly in the model.	Where said values are derived from external sources, the latter are typically referenced explicitly in Chapter 4 as well as in the accompanying script. Where they are assumed, this, too, is generally indicated in the script.

A.4 STEPS 3-4—PROCESSING THE DATA AND INCORPORATING UKPDS-OM2 SIMULATION RESULTS

The user now processes the inputs listed above to construct the transition probability and reward matrices, two central components of the decision problem. One achieves this by running several scripts in the order indicated in Table 4:

Table 4: Overview of data processing steps		
File name	Program	Function and prerequisites, if applicable
<i>NHS_2011_Calculations_042019.R</i>	R	Uses data from the NHS 2011 PUMF (Statistics Canada, 2014) and information regarding federal and provincial tax rates (EY, 2018) to tabulate household employment and retirement income. No prerequisites.

Table 4: Overview of data processing steps		
<i>Life_Tables_042019.R</i>	R	Uses data from the 2014-16 life tables for Manitoba (Statistics Canada, 2018), to calculate the likelihood of mortality by age and gender for healthy individuals. Also calculates life expectancy at birth (used in calculating the number of years of which people will continue drawing upon personal savings in retirement). No prerequisites.
<i>SLID_2011_Calculations_042019.R</i>	R	Drawing upon the 2011 SLID PUMF (Statistics Canada, 2014), calculates the probability of annual employment state transitions for individuals of varying age, gender, and level of educational attainment. No prerequisites.
<i>Presenteeism_Calculations_042019.R</i>	R	Drawing upon results presented in Lavigne et al. (2003), calculates monthly productivity losses attributable to T2DM by age, gender, ethnicity, and level of educational attainment. No prerequisites.
<i>DPoRT_Calculations_042019.R</i>	R	Drawing upon risk equations published by Rosella et al. (2014), calculates the probability of transitioning from healthy to T2DM for individuals varying by age, gender, ethnicity, level of educational attainment, and adherence to LTPA and healthy eating habits. User must first have executed the script entitled NHS_2011_Calculations_042019.R.
<i>UKPDS_Data_Template_V5.xlsm</i>	Visual Basic for Applications	Generates input sheets for use by the UKPDS-OM2 model. User must first have executed the script entitled DPoRT_Calculations_042019.R.
UKPDS-OM2 controller	Custom-built application	Calculates the likelihood of mortality, HRQoL, and workplace absenteeism. User must first have run the macro embedded in UKPDS_Data_Template_V5.xlsm, which is accessible through a button located on the user ribbon at the top of the screen under the “UKPDS Macro” tab.
<i>UKPDS_OM2_Results_Processing_042019.R</i>	R	Processes the results of the simulations implemented by the UKPDS-OM2 to facilitate integration into the decision model. User must first have carried out these simulations through the UKPDS-OM2 controller.

The scripts listed in Table 4 need only be executed once, as opposed to later scripts, which the user may wish to run several times (e.g., for the purposes of conducting sensitivity analyses). To

supplement the table, Figure 2 demonstrates precisely how the model processes and combines raw data from a wide range of sources to construct the inputs needed to set up the decision problem:

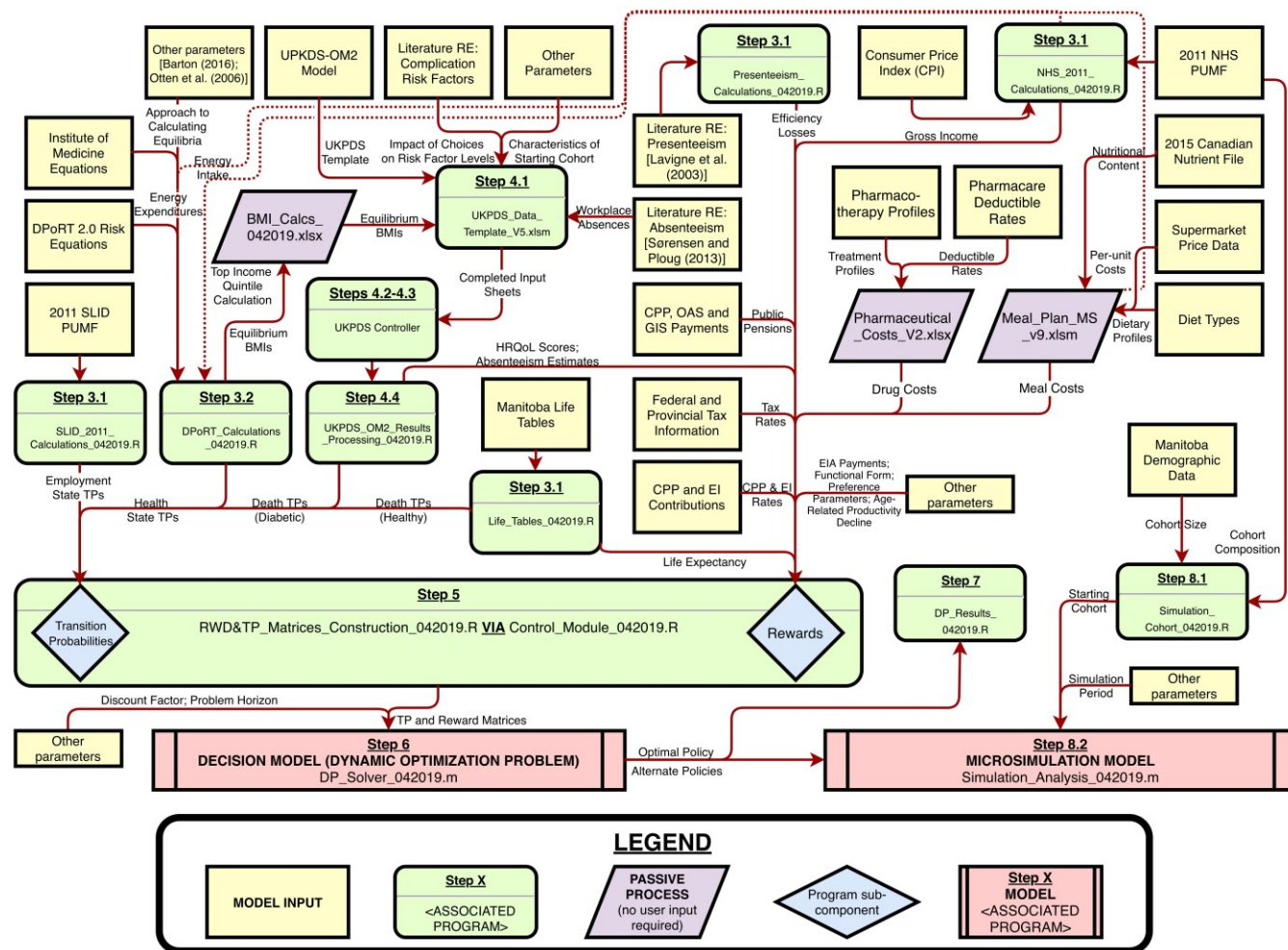


Figure 2: Flowchart illustrating the compilation of model inputs from raw data

It should be noted that whereas the UKPDS-OM2 data template and the R script for processing UKPDS-OM2 simulation results were both developed through this research, and are briefly discussed in Section 3.6.2 in Chapter 3 of the thesis as well as in Appendix D. However, the UKPDS-OM2 itself was created by the Diabetes Trials Unit and Health Economics Research Centre at the University of Oxford. I therefore refer the reader to user documentation prepared by those institutions for additional information regarding its use (University of Oxford Diabetes Trials Unit & Health Economics Research Centre, 2015).

A.5 STEP 5—GENERATING THE REWARD AND TRANSITION PROBABILITY MATRICES

The preceding steps culminate in the construction of the transition probability and reward matrices. Generation of these critical inputs for the baseline and sensitivity analyses is undertaken by executing *Control_Module_042019.R* and *Control_Module_SA_042019.R*, respectively, both of which are stored in the base directory, and which function by passing parameters to *RWD&TP_Matrices_Construction_042019.R* and extracting the results:

- ❖ Results for the baseline analysis are stored in the subfolders “RWD_Matrix_Components” and “TP_Matrix_Components” in C:\...\Diabetes_Model_042019\MATLAB\Baseline_Results\MATLAB_Output.
- ❖ Results for the sensitivity analysis are stored in similarly labelled subfolders in C:\...\Diabetes_Model_042019\MATLAB\Sensitivity_Analysis*\MATLAB_Output; in this case, however, the model saves separate sets of results for each parameter varied in the analysis (denoted by the asterisk).

For the baseline analysis, the model will by default produce transition probability and reward matrices for decision-makers with all 24 combinations of gender, ethnicity (i.e., Afro-Caribbean, Asian-Indian and Caucasian), and education (i.e., no certificate, diploma or degree, high school diploma or equivalent, post-secondary certificate or diploma below bachelor level, or university certificate, diploma, or degree at bachelor level or above) incorporated into the model.

For the sensitivity analysis the user must specify the characteristics of the decision-maker serving as the basis for the analysis.

A.6 STEPS 6-7—SOLVING THE DYNAMIC OPTIMIZATION PROBLEM AND INTERPRETING THE RESULTS

The user can now solve the decision model in MATLAB by initializing *DP_Solver_042019.m* or *DP_Solver_SA_042019.m* from the base directory ([C:\...\Diabetes_Model_042019](#)), which implement the baseline and sensitivity analyses, respectively. Both scripts employ backward induction techniques using the *mdpsolve* function packaged with the MDPSolve Toolbox. To function properly, the function requires the user to supply suitable reward and transition probability matrices (produced in the previous step) and specify the decision-maker's discount factor how far into the future the optimization problem extends (specified in the scripts themselves).

The decision model generates optimal policies and value functions for each combination of gender, ethnicity and level of education. These are stored as DAT files in separate folders (i.e., one for each person "type") within the following sub-directories:

- ❖ *Baseline analysis:* C:\...\Diabetes_Model_042019\MATLAB\Baseline_Results\MATLAB_Output.
- ❖ *Sensitivity analysis (the asterisk is a placeholder for sub-folders for each parameter varied in the analysis):* C:\...\Diabetes_Model_042019\MATLAB\Sensitivity_Analysis*\MATLAB_Output.

The model also includes two R scripts in the base directory which support interpretation of the solution to the dynamic optimization problem and replicate the summary tables presented in Section 3.2 in Chapter 4 of the thesis, as well as in Appendix F and Appendix G. *DP_Results_042019.R* generates results for the baseline analysis, while *DP_Results_SA_042019.R* and produces optimal policies and value functions for the sensitivity analysis.

Upon executing the backward induction algorithm, MATLAB will commonly warn the user that some rows of the transition probability matrix do not sum to 0. This appears attributable to a loss of precision occurring when matrix components are written into Excel from R. Although there is no straightforward remedy, the figures in question are extremely small (on the order of 10^{-9} or less) and are expected to have little or no impact on the model results.

In passing, I note that generating the transition probability and reward matrices and solving the dynamic optimization problem are computationally intensive operations that rely upon the application of a range of memory and performance management techniques, including the following:²

² Another option considered for memory and performance management included using single-precision floating point numbers in calculations, which require only half as much memory as double-precision floating point numbers). This was ultimately determined not to be viable, because sparse matrices (which are central to the construction of the transition probability matrices) work only with double or logical data (MathWorks, Inc., 2017b).

- ❖ *Preallocating memory to arrays.* Preallocation of memory to an array is often recommended as a means of improving performance, because MATLAB requires additional time to incrementally increase the size of a data structure (MathWorks, Inc., 2017a). This was accomplished by simply pre-specifying the size of each data structure and assigning zero values to each entry; subsequent calculations then replace these values as required.
- ❖ *Employing sparse arrays.* Sparse arrays are data structures that are mostly empty, storing only non-zero elements and their row indices (MathWorks, Inc., 2017b). When data density is sufficiently low,³ they can substantially reduce the amount of memory required for data storage, and may also improve performance (Altman, 2014, pp. 137–138). Sparse arrays are extremely useful within the context of the individual decision model, because the density of the transition probability matrix is approximately 1.31×10^{-5} .⁴
- ❖ *Applying matrix manipulation techniques.* The amount of memory required to store a sparse matrix depends upon both the number of non-zero entries and the number of matrix columns.⁵ For example, a sparse matrix consisting of a single row and one trillion columns would require a minimum of approximately 8,000 GB of memory, easily exceeding the working memory of the author's computer, while a sparse matrix consisting of one trillion rows and a single column would require a minimum of only 32 bytes. This suggests that it is preferable from a memory management standpoint to store sparse matrices in such a way that

³ In a full matrix, 8 bytes is required to hold a single double-precision floating point number in memory; thus, an $m \times n$ matrix requires $8 \times m \times n$ bytes of memory. By contrast, the minimum data storage requirement on a 64-bit system for a double $m \times n$ sparse matrix with nnz non-zero elements is as follows: $bytes = \max(nnz, 1) \times 16 + (n + 1) \times 8$, where n is the number of matrix columns (MathWorks, Inc., 2014). This demonstrates that sparse matrices require more memory to store each data entry, which is why they are less efficient than full matrices when data density is high.

⁴ The transition probability matrix is $228,357 \times 228,357$; however, there are only 685,065 possible transitions.

⁵ Refer to footnote 3 for a discussion of memory storage requirements for sparse arrays. Requirements for N-dimensional sparse arrays depend on the number of non-zero entries and the length of the last dimension of the array.

the last dimension is the shortest, recognizing that it is always possible to transpose or rearrange the matrix.

- ❖ *Vectorization (implicit parallelization)*. While it is often convenient to undertake repetitive operations by applying for-loops, this is often very time-consuming and can be prone to error. In many instances, it is possible to achieve dramatic improvements in performance through vectorization, which applies a specific operation to an entire array simultaneously, taking advantage of opportunities to carry out operations in parallel rather than sequentially (Altman, 2014, p. 223; MathWorks, Inc., 2018).

A.7 STEP 8—UNDERTAKING MICROSIMULATIONS

Before continuing, the user must solve the dynamic optimization problem using baseline parameter values by executing *DP_Solver_042019.m* in MATLAB from the base directory (i.e., [C:\...\Diabetes_Model_042019](#)). This generates the optimal transition matrices that drive the simulation model. In addition, they must define the characteristics of the patient cohort whose health and economic transitions they wish to study. The user can replicate the results presented in this thesis by executing *Simulation_Cohort_042019.R* in R (also located in the base directory).

With this complete, the user can conduct patient-level microsimulations in MATLAB by running *Simulation_Analysis_042019.m*. The figures presented in Section 5.3 in Chapter 4 of the thesis are stored in [C:\...\Diabetes_Model_042019\MATLAB\Simulation_Results\Figures](#), while detailed data tables containing the simulation results can be found in [C:\...\Diabetes_Model_042019\MATLAB\Simulation_Results\Tables](#).

NOTE: To promote reproducibility of results, the microsimulation program seeds MATLAB's random number generator using the same nonnegative integer seed for each run (i.e., '1234' with the Mersenne Twister generator algorithm). However, the user can easily disable this feature by modifying the value of the variable **Thesis_Results**, which, unless set equal to one, initializes the microsimulation using whatever the random seed may be on the user's machine at that time.

REFERENCES

Altman, Y. M. (2014). *Accelerating MATLAB Performance: 1001 tips to speed up MATLAB programs*. CRC Press.

Bache, S. M., & Wickham, H. (2014). magrittr: A Forward-Pipe Operator for R (Version 1.5). Retrieved from <https://CRAN.R-project.org/package=magrittr>

Dahl, D. B., Scott, D., Roosen, C., Magnusson, A., & Swinton, J. (2018). xtable: Export Tables to LaTeX or HTML (Version 1.8-3). Retrieved from <https://CRAN.R-project.org/package=xtable>

Diabetes Trials Unit. (2018, March 10). UKPDS Outcomes Model: Home Page. Retrieved February 21, 2015, from University of Oxford website: <http://www.dtu.ox.ac.uk/outcomesmodel/>

Dowle, M., & Srinivasan, A. (2019). data.table: Extension of `data.frame` (Version 1.12.2). Retrieved from <https://CRAN.R-project.org/package=data.table>

Dragulescu, A. A., & Arendt, C. (2018). xlsx: Read, Write, Format Excel 2007 and Excel 97/2000/XP/2003 Files (Version 0.6.1). Retrieved from <https://CRAN.R-project.org/package=xlsx>

EY. (2018, June 22). *Manitoba: Combined federal and provincial personal income tax rates, 2018*. Retrieved from [https://www.ey.com/Publication/vwLUAssets/Tax-Rates-Manitoba-2018/\\$FILE/Tax-Rates-Manitoba-2018.pdf](https://www.ey.com/Publication/vwLUAssets/Tax-Rates-Manitoba-2018/$FILE/Tax-Rates-Manitoba-2018.pdf)

Fackler, P. L. (2014, January 1). MDPSolve: Software for Dynamic Optimization. Retrieved April 24, 2018, from MDPSolve: Software for Dynamic Optimization website:
<https://sites.google.com/site/mdpsolve/>

Gordon, M., Gragg, S., & Konings, P. (2019). htmlTable: Advanced Tables for Markdown/HTML (Version 1.13.1). Retrieved from <https://CRAN.R-project.org/package=htmlTable>

Grolemund, G., & Wickham, H. (2011). Dates and Times Made Easy with lubridate. *Journal of Statistical Software*, 40(3), 1–25.

Grothendieck, G. (2018). gsubfn: Utilities for Strings and Function Arguments (Version 0.7). Retrieved from <https://CRAN.R-project.org/package=gsubfn>

Henry, L., & Wickham, H. (2019). purrr: Functional Programming Tools (Version 0.3.2). Retrieved from <https://CRAN.R-project.org/package=purrr>

Lavigne, J. E., Phelps, C. E., Mushlin, A., & Lednar, W. M. (2003). Reductions in individual work productivity associated with type 2 diabetes mellitus. *Pharmacoeconomics*, 21(15), 1123–1134.

Lugo, M. (2018). CANSIM2R: Directly Extracts Complete CANSIM Data Tables (Version 1.14.1). Retrieved from <https://CRAN.R-project.org/package=CANSIM2R>

MathWorks, Inc. (2014, April 21). Memory usage in sparse matrix: MATLAB Answers: MATLAB Central. Retrieved December 24, 2017, from MathWorks website:

<https://www.mathworks.com/matlabcentral/answers/126295-memory-usage-in-sparse-matrix>

MathWorks, Inc. (2017a, December 24). Preallocation: MATLAB & Simulink. Retrieved December 24, 2017, from MathWorks website:

https://www.mathworks.com/help/matlab/matlab_prog/preallocating-arrays.html

MathWorks, Inc. (2017b, December 24). Sparse Matrices: MATLAB & Simulink. Retrieved December 24, 2017, from MathWorks website:

<https://www.mathworks.com/help/matlab/sparse-matrices.html>

MathWorks, Inc. (2018, November 1). Vectorization: MATLAB & Simulink. Retrieved November 1, 2018, from MathWorks website:

https://www.mathworks.com/help/matlab/matlab_prog/vectorization.html

Müller, K. (2017). here: A Simpler Way to Find Your Files (Version 0.1). Retrieved from <https://CRAN.R-project.org/package=here>

Müller, K., & Wickham, H. (2019). tibble: Simple Data Frames (Version 2.1.1). Retrieved from <https://CRAN.R-project.org/package=tibble>

Ooms, J. (2014). The jsonlite Package: A Practical and Consistent Mapping Between JSON Data and R Objects. *ArXiv:1403.2805 [Stat.CO]*. Retrieved from <https://arxiv.org/abs/1403.2805>

Ooms, J. (2019). pdftools: Text Extraction, Rendering and Converting of PDF Documents (Version 2.2) [Purr]. Retrieved from <https://CRAN.R-project.org/package=pdfutils>

Ooms, J., Lang, D. T., & Hilaiel, L. (2018). jsonlite: A Robust, High Performance JSON Parser and Generator for R (Version 1.6). Retrieved from <https://CRAN.R-project.org/package=jsonlite>

Ren, K., & Russell, K. (2016). formattable: Create “Formattable” Data Structures (Version 0.2.0.1). Retrieved from <https://CRAN.R-project.org/package=formattable>

Rosella, L. C., Lebenbaum, M., Li, Y., Wang, J., & Manuel, D. G. (2014). Risk distribution and its influence on the population targets for diabetes prevention. *Preventive Medicine*, 58(Supplement C), 17–21. <https://doi.org/10.1016/j.ypmed.2013.10.007>

Spinu, V., Grolemund, G., & Wickham, H. (2018). lubridate: Make Dealing with Dates a Little Easier (Version 1.7.4) [Magr]. Retrieved from <https://CRAN.R-project.org/package=lubridate>

Statistics Canada. (2014, July 29). *2011 National Household Survey Public-Use Microdata File (PUMF)*. Retrieved from <https://www150.statcan.gc.ca/n1/en/catalogue/99M0001X2011001>

Statistics Canada. (2018, June 28). Life Tables, Canada, Provinces and Territories, 1980/1982 to 2014/2016. Retrieved September 20, 2018, from Statistics Canada website: <https://www150.statcan.gc.ca/n1/en/catalogue/84-537-X2018002>

Stodden, V., Leisch, F., & Peng, R. D. (Eds.). (2014). *Implementing reproducible research*. Boca Raton: CRC Press, Taylor & Francis Group.

University of Manitoba. (2018, November 17). University of Manitoba—Libraries—About MSpace. Retrieved November 17, 2018, from University of Manitoba website:
<http://umanitoba.ca/libraries/elibrary/mspace/>

University of Oxford Diabetes Trials Unit, & Health Economics Research Centre. (2015, May 11). *UKPDS Outcomes Model User Manual*. Retrieved from
<https://www.dtu.ox.ac.uk/outcomesmodel/OM2Manual.pdf>

Walker, A. (2018). openxlsx: Read, Write and Edit XLSX Files (Version 4.1.0). Retrieved from
<https://CRAN.R-project.org/package=openxlsx>

Wickham, H. (2007). Reshaping Data with the reshape Package. *Journal of Statistical Software*, 21(12), 1–20.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Retrieved from
<http://ggplot2.org>

Wickham, H. (2017). Reshape2: Flexibly Reshape Data: A Reboot of the Reshape Package (Version 1.4.3). Retrieved from <https://CRAN.R-project.org/package=reshape2>

Wickham, H. (2018). httr: Tools for Working with URLs and HTTP (Version 1.4.0). Retrieved from <https://CRAN.R-project.org/package=httr>

Wickham, H. (2019a). rvest: Easily Harvest (Scrape) Web Pages (Version 0.3.3). Retrieved from
<https://CRAN.R-project.org/package=rvest>

Wickham, H. (2019b). stringr: Simple, Consistent Wrappers for Common String Operations (Version 1.4.0). Retrieved from <https://CRAN.R-project.org/package=stringr>

Wickham, H., & Bryan, J. (2019). readxl: Read Excel Files (Version 1.3.1). Retrieved from <https://CRAN.R-project.org/package=readxl>

Wickham, H., Chang, W., Henry, L., Pedersen, T. L., Takahashi, K., Wilke, C., ... RStudio. (2019). ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics (Version 3.1.1). Retrieved from <https://CRAN.R-project.org/package=ggplot2>

Wickham, H., & Henry, L. (2019). tidyr: Easily Tidy Data with “spread()” and “gather()” Functions (Version 0.8.3). Retrieved from <https://CRAN.R-project.org/package=tidyr>

Xie, Y. (2015). *Dynamic documents with R and Knitr* (Second edition). Boca Raton: CRC Press/Taylor & Francis.

Xie, Y. (2019). knitr: A General-Purpose Package for Dynamic Report Generation in R (Version 1.22). Retrieved from <https://CRAN.R-project.org/package=knitr>