Pension Simulation Calculator

By

Priya Balachandran Mary

An Project Report submitted to

University at Albany

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Computer Science and Engineering

Prof. Petko Bogdanov

Date

AUTHORIZATION FOR REPRODUCTION
OF PROJECT

I grant permission for the reproduction of this project in its entirety, without further authorization from me, on the condition that the person or agency requesting reproduction, absorb the cost and provide proper acknowledgment of authorship.

Date_____ _____

First Name Last Name

Street Address

City, State, Zip

ABSTRACT

In the United States, funding for most government pension plans is based on assumptions for a combination of actuarial inputs like inflation rate, salary growth rate, discount rate, cost methods, mortality/longevity assumptions etc. Sometimes these inputs are overly optimistic and predict lower liabilities only to make room for liabilities (unfunded) that are beyond a plan's then estimated competency.

With the Pension Simulation Calculator, we wish to study the effect of changes to actuarial inputs on a plan's costs and liabilities for different types of simulated pension plan population profiles and assets.

The Pension Simulation Calculator provides an R Shiny user interface to accept different actuarial inputs and generate a synthetic population to resemble a pension plan population profile of interest to the end user, a dataset for the plan costs and liabilities for different actuarial inputs.

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

Pension plans made by US state and local government take into consideration sets of values for actuarial inputs, which may seem to promise lower liabilities and appear to be a well funded plan at the time. The impact of these decisions in the long term cannot be constant since funding a plan has to take into consideration a combination of many constantly changing actuarial inputs. Small changes to these inputs or a combination of these inputs could give us a 100 percent funded plan or a very poorly funded plan. Changes to these inputs may seem to be cause only a small marginal change for the current year but may bring up liabilities drastically if the effect these changes are not analyzed for years to come.

The Pension Simulation Calculator, is a Shiny application based on Dr. Chen and Dr. Matkin's '*The Relationship between Actuarial Inputs and the Financial Condition of Public Pensions*' [1], built to study the effect of the combination of actuarial inputs and aims to provide help make better funding decisions for a pension plan.

2 SYSTEM DESIGN

The Pension Simulation Calculator comprises of the following components:

2.1 SHINY APPLICATION

The Pension Simulation Calculator is build using a package offered by RStudio for web development called the 'shiny' package. The package allows us to create a user interface through a set of predefined R functions to create UI components, which internally knits HTML5, some JavaScript and CSS3 code together in the "ui.R" file as shown in Figure 1. The model uses the inputs to these UI components and passes it down to the "server.R" which acts like a controller and makes function calls by to the "functions.R" file comprises of the model's logic for synthetic population generation and dataset generation. The server then receives the data generated from these function calls and dynamically creates plots, data tables and in some cases dynamic UI and returns these back to the ui.R.

2.2 SHINY SERVER

The Shiny server component hosts the shiny application/ applications online. It is a precompiled binary provided by R studio, which sets up a web server and allows us to

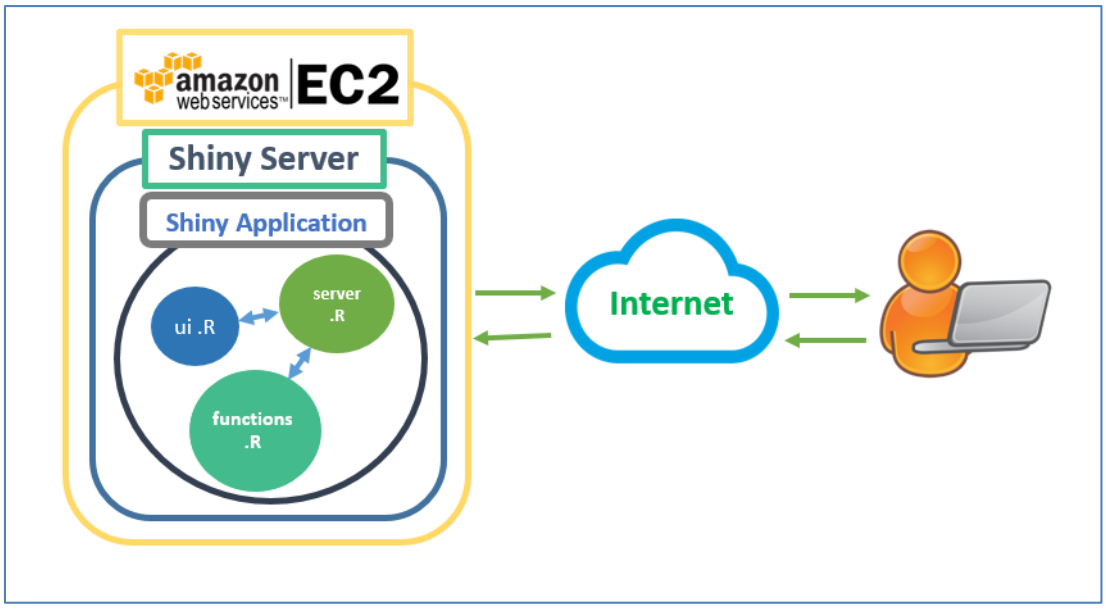

Figure 1: System Design

host shiny applications with different URLs or different ports. It uses the AWS EC2 instance's available ports and defines a configuration file for launching the Shiny Application.

2.3 AMAZON WEB SERVICES EC2 INSTANCE

The AWS EC2 instance is where we place our Shiny application and install the Shiny server. The AWS instance runs on Ubuntu 12.04 and we install the precompiled Shiny server binary for Ubuntu 12.04. The site is available to the users through the AWS URL. The user passes requests to the AWS instance, which passed on to the Shiny Server and the Shiny application. For hosting the model, we use an amazon machine image, 'RStudio-0.99.491_R-3.2.3_ubuntu-14.04-LTS-64bit -ami-7f9dc615' [2] which is a ready to run RStudio server.

3 IMPLEMENTATION

3.1 PREREQUISITES

To run the model the following binaries and packages are used. The Model was initially build on a Windows Version 10.0.10586 system and later deployed on an AWS EC2 Ubuntu instance for accessing the calculator online.

1. R Version 3.2.5 for Windows, R Version 3.2.0 for Ubuntu
2. RStudio Version 0.99.484 for Windows
3. R Packages:
    a. Scales – a package that maps data with aesthetics for better visualization and frequently used in the model for number formatting [3].
    b. BayesBridge – a package for Bayesian Bridge Regression used in the model to create synthetic distributions for population [4].
    c. Shiny – a package to create interactive web application in R [5].
    d. Shinythemes – a package to embed different bootstrap themes to the web application [6].
    e. Plotly – an interactive, browser-based charting library built on a JavaScript graphing library called plotly.js [7].
    f. doParallel – an adaptor for the in built 'parallel' package to enable the use of multiple cores through a for each style loop [8].
    g. foreach – a package to enable for each style looping in R [9].
    h. XLConnect – a package to read and write xls sheets, used in the model for importing mortality tables, termination rates etc [10].
4. Shiny Server v1.4.2.786 Open Source for Ubuntu 12.04 and later

3.2 SHINY SERVER SETUP

1. After we set up the AWS image for RStudio server we update the system with the latest version of R and Shiny [11] using the following commands:

    $sudo apt-get install r-base

$sudo su - \-c“ R – E

\”install.packages(‘shiny’,repos=’https://cran.rstudio.com/')\""

2. We update the gdebi tool, later used to install shiny server, using the following command:

   $ sudo apt-get install gdebi-core

3. We download the Shiny server binary from rstudio.org and install shiny server using the following commands

   $ wget https://download3.rstudio.org/ubuntu-12.04/x86_64/shiny-server-1.4.2.786amd64.deb

   $ sudo gdebi shiny-server-1.4.2.786-amd64.deb

4. We later change the shiny server configuration file to point to the shiny application directory in the AWS instance and configure the port for the website.

4 THE MODEL

The model consists of four sections with the first three sections actively receiving user input and input changes, generating population data, Normal costs, AAL and summarized cost of assets and liabilities for different age groups in the plan's population. The last section shows impact of changes in discount rate and salary growth rate on funding ratio and actuarial required contribution.

4.1 POPULATION & PLAN SETTINGS

In this section, the user has to provide inputs for a plan's population and inputs for the plan's feature.

Population data: The inputs to the population generation function are the number of employees in the plan, typical entry age of an employee in the plan, retirement age and the age distribution of the active employees as shown in the 'Who participates in the plan?' section in Figure 2. These inputs are inputs to the rtnorm function of the Bayesian Bridge package mentioned in section 3.1.

For option 'every age group has a relatively the same number of workers', members are equally distributed over each age group.

For option 'a large portion of current employees are nearing retirement' the inputs are passed to the rtnorm function as shown below where n is the number of participants, retire represents the retirement age, sig represents the standard deviation, left and right parameters define the range for the different age groups participating in the plan

rtnorm(n, mean= retire, sig=15, left = entry_age ,right = retire)

Similarly, for option 'a small portion of current employees are nearing retirement', the rtnorm function has the same inputs except that the mean set to the entry age since the plan's population has a higher number of younger participants.

For the last option, 'Customize the age distribution' we allow the user to choose a mean age between the entry age and retirement age to distribute the population. At this point the server generates a dynamic slider component for the user to change the mean age

Retirees: Retirees are added to the plan based on the number of participants at retirement. A mortality table of choice is then applied to the number of participants at retirement and the number of retirees decrease as the distribution reaches age 100 (the model's maximum age).

The frequencies for the two populations (active and retirees) are then added as two separate traces to the 'plot_ly' function from the plotly package and rendered by the Shiny server to form the age distribution graph in Figure 2.

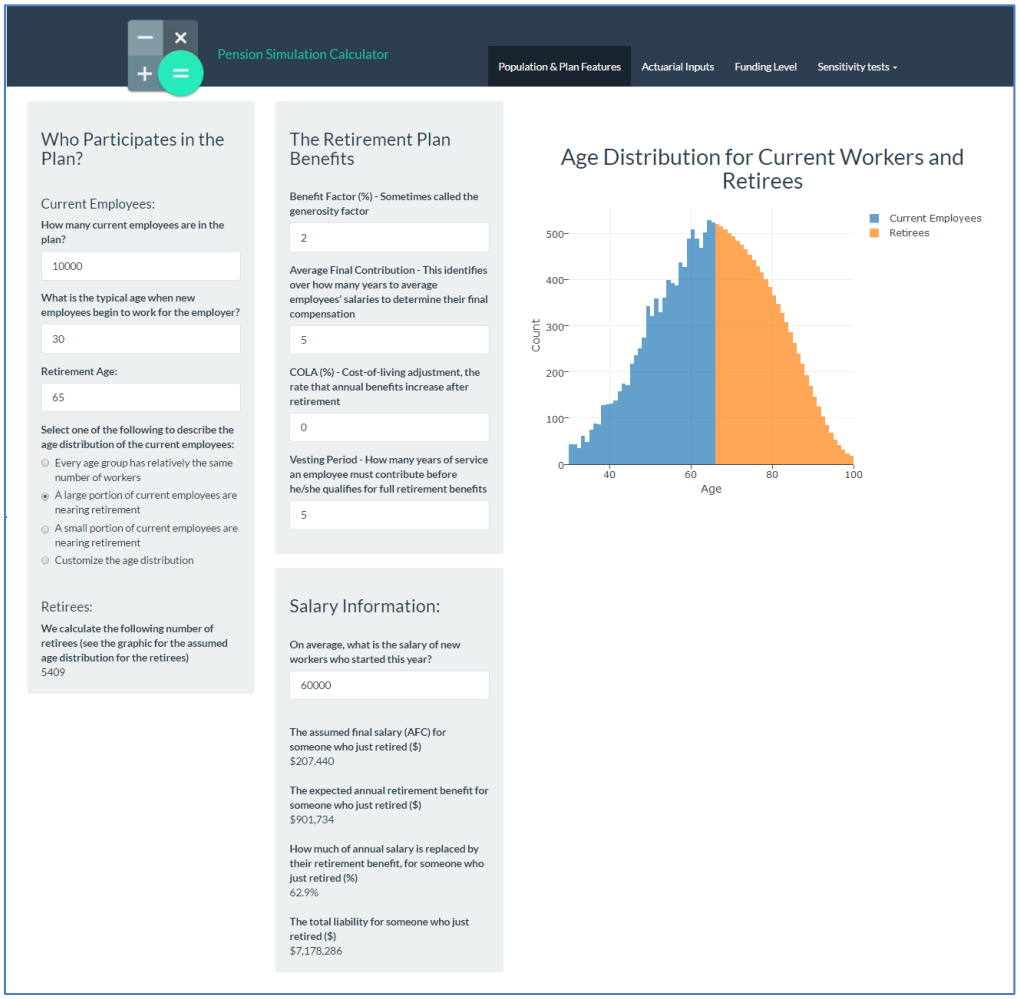

Figure 2: Population and Plan Settings

Retirement Plan Benefits: This section takes in benefit factor, average final contribution, cost of living adjustments and vesting period, which acts as inputs for the calculations in the next section and helps set the plan features.

Salary Information: This section allows the user to input a person's entry salary and uses the default values for the actuarial inputs in the next section to calculate a person's salary at retirement, retirement annuity, replacement rate and present value of the annuity for a plan.

4.2 Actuarial Inputs

This section of the model accepts the cost method, discount rate, expected salary growth rate, amortization period and the mortality table, using which we compute the normal cost for each individual an the plan and the AAL associated with them.

The graphs in this section (Figure 3) shows the growth in one person's normal cost and AAL from hire to retirement. The Normal cost/Payroll for an individual will be constant given the underlying assumption of one salary growth rate and one discount growth rate from hire to retirement.

Figure 3: Actuarial Inputs

Normal cost and AAL are calculated using the following formulas:

$$ {}^{r}(NC)_{x}^{EAN} = \frac{s_x \ {}_{x-y}P_y \ v^{x-y}}{s_y \ {}^{s}\ddot{a}_{y:\overline{r-y|}}} * ({}^{r}(PVFB)_x + {}^{r}(PVTC)_x) $$

$$ {}^{r}(NC)_{x}^{PUC} = \frac{1}{r-y} * ({}^{r}(PVFB)_x + {}^{r}(PVTC)_x) $$

$$ {}^{r}(AAL)_{x}^{EAN} = ({}^{r}(PVFB)_x + {}^{r}(PVTC)_x) \ - ({}^{r}(NC)_{x}^{EAN} * \frac{{}_{x-y}P_y \ v^{x-y}}{{}^{s}\ddot{a}_{y:\overline{r-y|}}}) $$

$$ {}^{r}(AAL)_{x}^{PUC} = \frac{x-y}{r-y}({}^{r}(PVFB)_x + {}^{r}(PVTC)_x) $$

$$ Funding\ ratio_t = \frac{Asset_t}{AAL_t} $$

where,

${}^{r}(NC)_{x}^{EAN}$ is the normal cost at the age of *x* with the assumed retirement age of *r* under EAN method.

${}^{r}(NC)_{x}^{PUC}$ is the normal cost at the age of *x* with the assumed retirement age of *r* under PUC method.

${}^{r}(AAL)_{x}^{EAN}$ is the AAL at the age of *x* with the assumed retirement age of *r* under EAN method.

${}^{r}(AAL)_{x}^{PUC}$ is the AAL at the age of *x* with the assumed retirement age of *r* under PUC method.

x is the current age, which varies according to different employees at a certain year.

y is the entry age

r is the retirement age

${}^{r}(PVFB)_x$ is the present value of all future benefits at the age of x with the assumed retirement age of *r*

${}^{r}(PVTC)_x$ is the present value of all future termination costs at the age of x with the assumed retirement age of *r*

s_x is the salary at the age of *x*.

s_y is the salary at the entry age *y*.

${}_{x-y}P_y$ is the probability of surviving (*x*-*y*) years from the entry age *y* to the current age *x*.

v^{x-y} is the discount factor from entry age *y* to the current age *x*.

${}^{s}\ddot{a}_{y:\overline{r-y|}}$ is a salary-based annuity factor, which represents the present value of an employee's future salary from the entry age *y* to retirement age *r*, per unit of salary at age *y*.

4.3 FUNDING LEVEL

This section allows the user to choose a funding level for the plan. We use this level to create assets for the pension plan since we have AAL from the previous section.

We calculate assets using the following formula:

$$Asset_t = Funding\ Ratio * AAL_t$$

where,

$Funding\ ratio_t$ is the funding ratio for a plan at t time period.

$Asset_t$ is the asset value for a plan at t time period. The value of the asset is initially fixed as a defined portion of the liability. We keep the asset value fixed at t time periods.

This section also shows a summary of the plan's costs incurred and an age group wise distribution of Salary, Normal Cost and AAL.

4.4 SENSITIVITY TESTS

The sensitivity tests shows the effect of different discount rates and salary growth rates on funding ratio and ARC under different mortality tables and cost methods chosen in the actuarial inputs section 3.3.2 and population profiles created in 3.3.1.

Discount Rate: This section shows how the funding ratio and ARC changes as the discount rate changes keeping all other actuarial inputs constant. The graphs in Figure 4

show the trend in the funding ratio of a plan and ARC required for both cost methods as the discount rate increases and decreases.

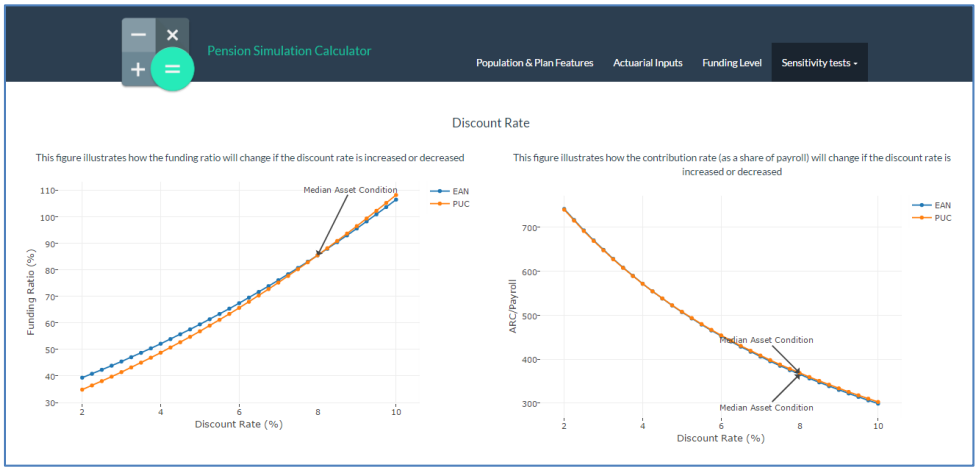

Figure 4: Sensitivity test to study the impact of different discount rates on funding ratio and ARC

<u>Salary Growth Rate</u>: This section shows how the funding ratio and ARC changes as the salary growth rate changes keeping all other actuarial inputs constant. Similar to the discount rate section, the graphs in Figure 5 show the trend in the funding ratio of a plan and ARC required for both cost methods as the salary growth rate increases or decreases.

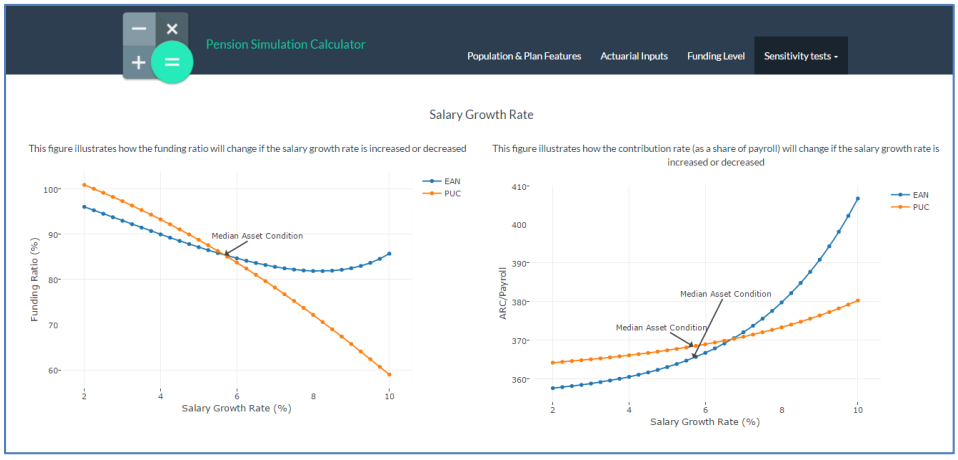

Figure 5: Sensitivity test to study the impact of different salary growth rates on funding ratio and ARC

The ARC and Funding ratio are calculated using the following formulas:

$$Funding\ ratio_t = \frac{Asset_t}{AAL_t}$$

$$ARC_t = NC_t + \frac{P}{r-g}\left(1 - \left(\frac{1+g}{1+r}\right)^n\right)$$

where,

$Funding\ ratio_t$ is the funding ratio for a plan at *t* time period.

$Asset_t$ is the asset value for a plan at *t* time period.

ARC_t is the annual required contribution at *t* time period for a plan.

NC_t is the normal cost at *t* time period for a plan for EAN or PUC.

P is the first payment for a participant

r is the real rate of return/discount rate

g is the payroll growth rate

n is years of service

5 CONCLUSION

With the current model, we have been able to calculate plan costs and see trends in funding ratio and ARC as we alter actuarial inputs and the population data.

The current model generates a dataset of normal costs, AAL, ARC, UAAL for a set of combinations of actuarial input values.

The model can be further extended to provide a time series analysis on the effect of marginal changes in these inputs over n years.

We utilize the doParallel package to get all cores in to make the dataset, however the dataset generation is still a long process for which employing more cores would be ideal. This is a limitation and is open for improvement.

6 REFERENCES

[1] *The Relationship between Actuarial Inputs and the Financial Condition of Public* Pensions - Dr. Gang Chen, Dr. David Matkin

[2] *RStudio Server Amazon Machine Image (AMI) http://www.louisaslett.com/RStudio_AMI/*

[3] *Scales R package https://cran.r-project.org/web/packages/scales/scales.pdf*

[4] *BayesBridge R package https://cran.r-project.org/web/packages/BayesBridge/BayesBridge.pdf*

[5] *Shiny home page http://shiny.rstudio.com/*

[6] *Shinythemes R package documentation, https://cran.r-project.org/web/packages/shinythemes/shinythemes.pdf*

[7] *plotly reference page for r https://plot.ly/r/*

[8] *Getting Started with doParallel and foreach https://cran.r-project.org/web/packages/doParallel/vignettes/gettingstartedParallel.pdf*

[9] *Using The foreach Package https://cran.r-project.org/web/packages/foreach/vignettes/foreach.pdf*

[10*] XLConnect R package https://cran.r-project.org/web/packages/XLConnect/vignettes/XLConnect.pdf*

[11] *https://cran.rstudio.com mirror for r base*