# Calendar Year Effect - Methodology

*Aniketh Pittea*

The document provides the methodology I use to illutrate calender year effects using stochastic inflation. I use stochastic inflation simply because it can be easily modelled using an AR(1) process, however the methodology can be easily adapted for other factors such as changes in Ogden rates.

**Step 1:**First, I generate future inflation rates using an AR(1) process. The error terms are assumed to follow a normal distribution with mean 0 and standard deviation 3.87%. I also assume that the average past inflation is 4.04% and use an autoregressive parameter of 0.6102 (the parameters I use are based on an economic scenario generator I recently developed. The paper is not published yet but has been submitted to the Annals of Actuarial Science for review).

The "chunk" below shows the code to generate 10 years of future inflation rates.

```
# AR(1) process to generate future inflation
rpi_sim=function(n=1,mu=0.0404,t=10,beta=0.6102,sigma=0.0387,initial=0.0387){
  error=matrix(data=NA, nrow=t, ncol=n)
  Y=matrix(data=NA, nrow=t, ncol=n)
  Z=matrix(data=NA,nrow=t, ncol=n)
  error[1,] <- 0
  Y[1,] <- 0
  Z[1,] <- initial
  for(j in 1:n){
    for(i in 2:t){
      error[i,j]=rnorm(1,0,sigma)
      Y[i,j]=beta*Y[i-1,j]+error[i,j]
      Z[i,j]=Y[i,j]+mu
    }
  }
  return(Z)
}


rpi <- matrix (data=NA,ncol=1,nrow=10)
rpi = rpi_sim(t=10)
rpi
```

```
##             [,1]
##  [1,] 0.03870000
##  [2,] 0.05566302
##  [3,] 0.08214445
##  [4,] 0.01342831
##  [5,] 0.03165801
##  [6,] 0.01653127
##  [7,] 0.07399885
##  [8,] 0.06542553
##  [9,] 0.13514441
## [10,] 0.02690574
```

**Step 2:** From the generated inflation rates, I create an inflation index

```
#Create an Inflation Index
rpi_index <- matrix(data=NA, nrow=10)
```

```
rpi_index[1] <- 1
for(i in 2:10){
  rpi_index[i] <- rpi_index[i-1]*(1+rpi[i])
}
rpi_index
```

```
##           [,1]
##  [1,] 1.000000
##  [2,] 1.055663
##  [3,] 1.142380
##  [4,] 1.157720
##  [5,] 1.194371
##  [6,] 1.214116
##  [7,] 1.303959
##  [8,] 1.389271
##  [9,] 1.577023
## [10,] 1.619454
```

**Step 3:** I load my data of projected incremental claims. I would obtain this from my Actuary-in-the-Box code. I do not show the code here, instead I load up some synthetic data.

```
#Load projected incremental claims data
pjic <- read.csv("C:\\Users\\Aniketh\\Dropbox\\Stochastic Reserving\\Calender_Year\\DataPJIC.csv",heade
pjic
```

```
##       V1        V2      V3       V4       V5        V6       V7       V8        V9
## 1     NA        NA      NA       NA       NA        NA       NA       NA  60207.69
## 2     NA        NA      NA       NA       NA        NA       NA 249436.9  47641.22
## 3     NA        NA      NA       NA       NA        NA 200530.9 265149.0  64020.37
## 4     NA        NA      NA       NA       NA  378493.3 189063.3 327723.2  53145.34
## 5     NA        NA      NA       NA 143618.7  618315.1 241213.9 320581.0 100695.86
## 6     NA        NA      NA 868899.0 472578.0  510835.9 384971.1 456090.6  85188.50
## 7     NA        NA 1597061 697027.8 960571.6  665348.5 386173.9 447554.6  72744.71
## 8     NA 1282537.1 1751571 758719.9 765716.1 1035183.9 339849.7 445817.8  86669.08
## 9 839826  902285.2 1157752 527510.7 659667.2  326312.2 271926.2 224868.5  79929.56
```

**Step 4:** I then apply my inflation index to my projected incremental claims.

```
#Projected Incrematal Claim Inflation Adjusted
pjic_inf <- matrix(data=NA,nrow=9,ncol=9)
for(i in 1:9){
for(j in 1:9){
pjic_inf[i,j] <- is.na(pjic[i,j])
if(pjic_inf[i,j]==TRUE){
  pjic_inf[i,j]=NA
}
else{
  pjic_inf[i,j]=pjic[i,j]*rpi_index[i+j-(9-1)]
}
 }
}
pjic_inf
```

```
##           [,1]    [,2]    [,3]     [,4]      [,5]      [,6]     [,7]     [,8]      [,9]
##  [1,]       NA      NA      NA       NA        NA        NA       NA       NA  63559.04
##  [2,]       NA      NA      NA       NA        NA        NA       NA 263321.3  54424.37
```

```
##  [3,]       NA      NA      NA       NA        NA        NA 211693.0 302900.9  74117.67
##  [4,]       NA      NA      NA       NA        NA  399561.4 215982.1 379411.8  63475.26
##  [5,]       NA      NA      NA       NA  151613.0  706350.7 279258.2 382892.7 122256.43
##  [6,]       NA      NA      NA 917264.6  539863.6  591405.0 459798.4 553746.8 111082.30
##  [7,]       NA      NA 1685959 796270.5 1112073.1  794673.1 468859.7 583592.8 101062.12
##  [8,]       NA 1353927 2000960 878385.3  914549.3 1256833.0 443150.1 619361.8 136679.15
##  [9,] 886573.2 1030752 1340353 630043.6  800912.3  425497.7 377779.1 354622.9 129442.27
```

Using this, I would then compare my Ultimo Ultimate and one-year Ultimate. I show the results/graphs in the document "Output".