Report - Aug 1st

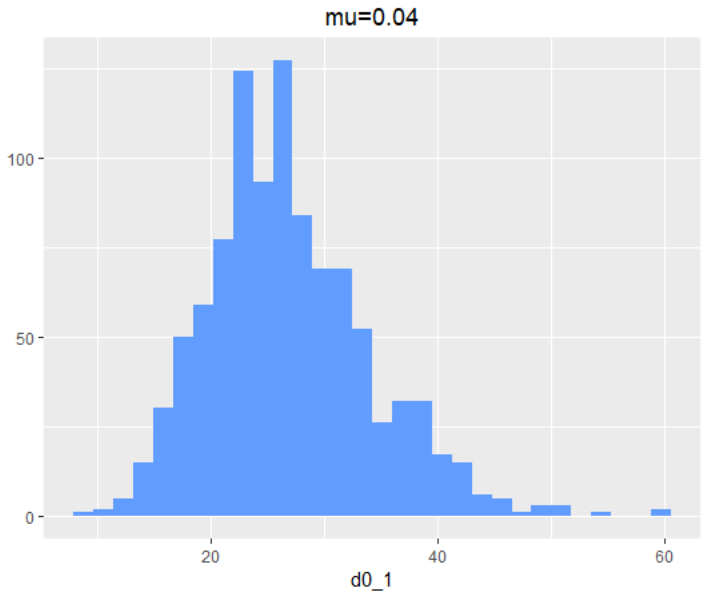
Tiancheng Hou

Note: Initial asset, b0, is set as 500

1. Method I: No Recalibration

mu=0.04

distribution of all 1000 d0




```
> summary(d0_1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  9.178  22.062  26.036  26.867  31.025  60.102
```

0.05 quantile:

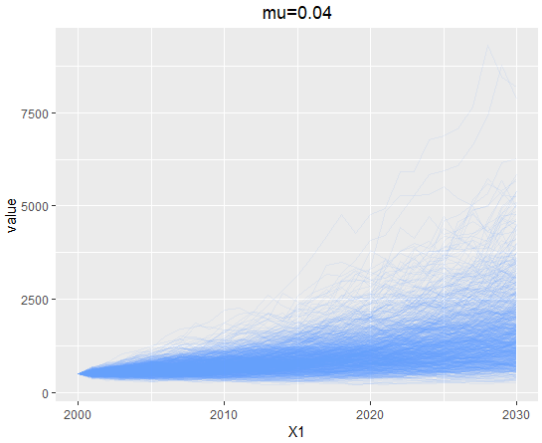
```
> mylist[[1]]$Withdrwal
 [1] 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628
[13] 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628
[25] 16.55628 16.55628 16.55628 16.55628 16.55628 16.55628
```

Ruin probability:

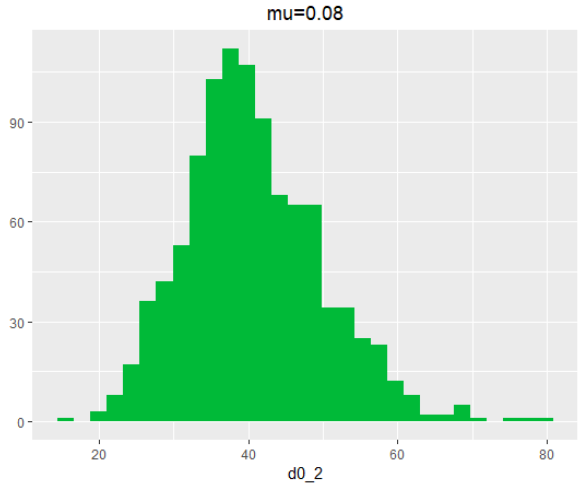
```
> sum(a)/1000
[1] 0.049
```

It should be exactly 0.05, and I get 0.049. I think it is because, for the 50th smallest d0, it generates a very small ending value, though very close to 0, slightly greater than 0. (?)

Path of price (Initial price is 500):



mu=0.08




```
> summary(d0_2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  16.40   34.62   39.71   40.66   46.41   80.61
```

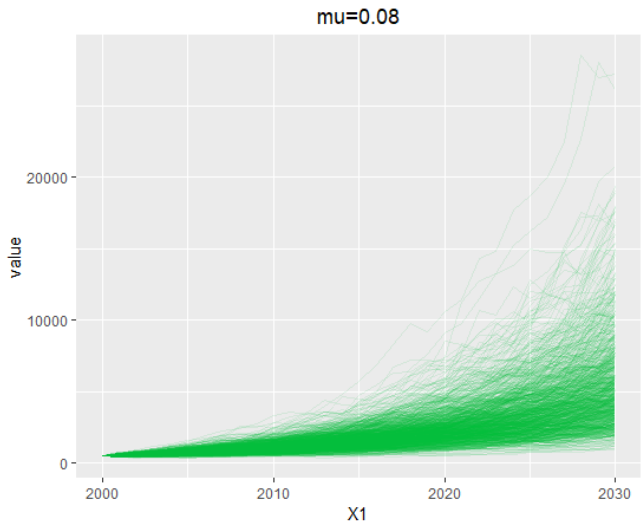
0.05 quantile:

```
mylist[[1]]$withdrwal

 [1] 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839

[13] 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839

[25] 27.09839 27.09839 27.09839 27.09839 27.09839 27.09839
```

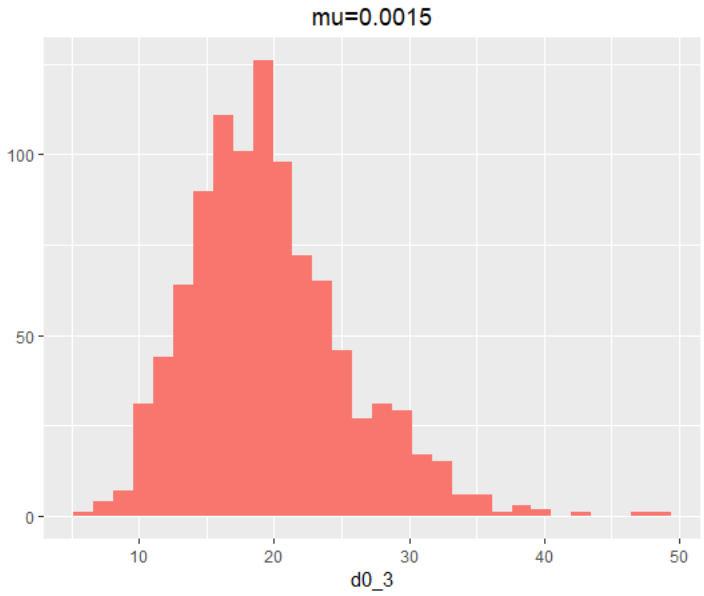
Ruin probability:

```
> sum(a)/1000
[1] 0.573
```

Path of price (Initial price is 500)



mu=0.015




```
> summary(d0_3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  6.056  15.649  18.941  19.654  22.879  48.836
```

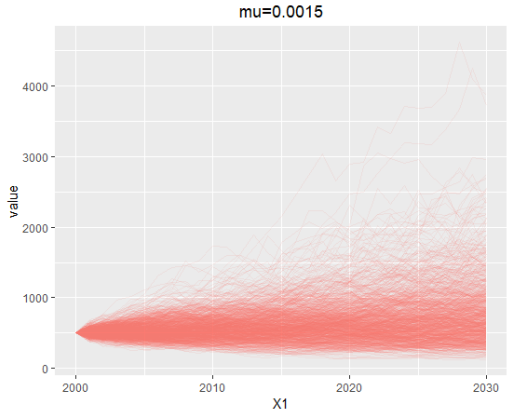
0.05 quantile:

```
> mylist[[1]]$Withdrwal
 [1] 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182
[13] 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182
[25] 11.50182 11.50182 11.50182 11.50182 11.50182 11.50182
```

Ruin probability:

```
> sum(a)/1000
[1] 0.003
```

Path of price (Initial price is 500)



2. Method II: Kalman Filter Recalibration

b.

Note: Initial mu and sigma are randomly set as 0.04 and 0.2, since we do not know the true value of them.

Result:

```
> optimal_kf
$`Withdrwal`
 [1] 23.28553 27.56753 22.93079 37.09878 54.77432 61.91534 48.24402 34.89147 37.18334 35.37883 34.24792 38.77763
[13] 37.47057 38.91209 41.66017 46.15855 44.42080 50.04995 36.89462 34.17286 33.44282 34.88557 36.81196 35.37293
[25] 37.95235 36.19516 43.31822 45.79551 41.03983 43.74838

$Remaining_value
 [1] 471.67800 520.50877 575.14433 642.82535 709.35047 565.12214 410.16770 439.66979 417.74525 393.79844 437.61164
[12] 412.43993 412.21058 422.94969 448.97992 409.70164 445.52915 316.63001 278.83721 255.90341 248.10453 240.73728
[23] 208.68863 197.75838 161.42315 158.93471 130.09573  80.03129  43.74838   0.00000
```

The first vector is the withdrawal amount in each year. The second vector is the remaining value of the founding in each year. **(Note: Initial asset, b0, is set as 500)**

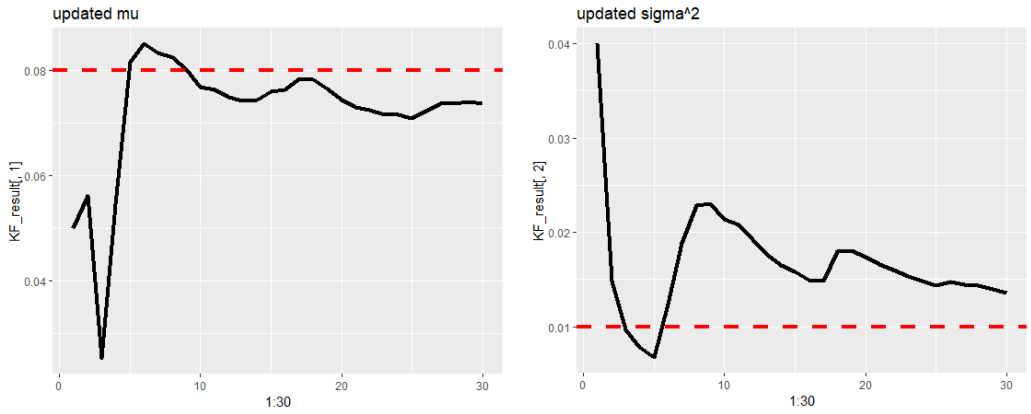
Mean of withdrawal vector is

```
> mean(optimal_kf$Withdrwal)
[1] 39.15326
```

,

which is greater than d0(23.28553). It means that K-F recalibration can reflect the market performance.

Update mu and sigma^2 with Kalman filter:



From above, we notice that even we set mu and sigma that is not very close to the “actual value”, they will converge to the it, as Kalman filter gathering more data.

c.

Result:

```
> optimal_kf
$`Withdrwal`
 [1] 26.33398 24.38049 14.57325 20.01644 27.67902 30.08672 22.54653 15.72576 16.21313 14.92780 14.10588 15.66889
[13] 14.93397 15.28293 16.23366 17.88514 17.11876 19.20607 14.09040 12.97561 12.66250 13.17617 13.88065 13.32303
[25] 14.28137 13.61115 16.28568 17.20832 15.41554 16.43117

$Remaining_value
 [1] 439.16768 455.54704 477.63850 512.33233 547.83262 423.48067 298.13791 310.04863 285.78233 261.39737 282.03888
[12] 258.12180 250.51633 249.74817 257.66581 228.52842 241.62671 166.86851 142.70372 127.14823 119.64756 112.64744
[23]  94.71297  87.02390  68.85443  65.70035  52.10387  31.04360  16.43117   0.00000
```

The first vector is the withdrawal amount in each year. The second vector is the remaining value of the founding in each year. **(Note: Initial asset, b0, is set as 500)**

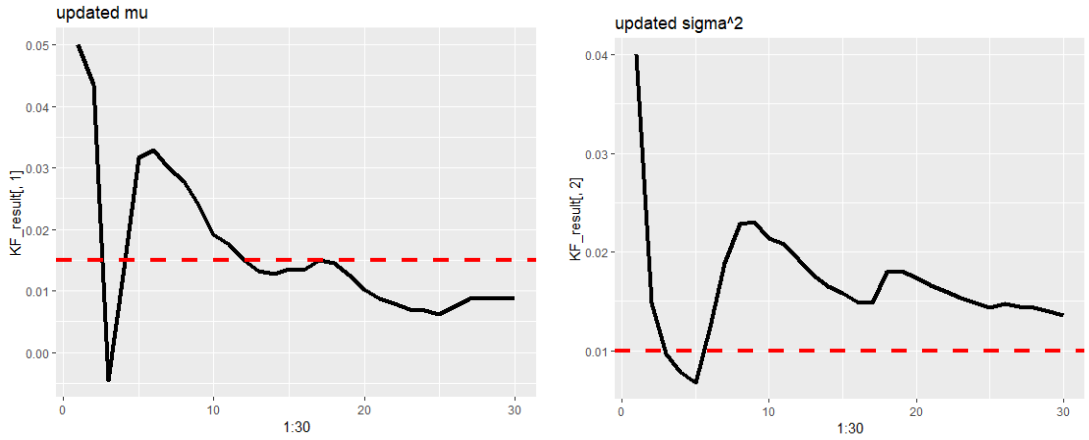
Mean of withdrawal vector is

```
> mean(optimal_kf$Withdrwal)
[1] 17.20867
```

,

which is smaller than d0(26.33398). It means that K-F recalibration can reflect the market performance.

Update mu and sigma^2 with Kalman filter:



From above, we notice that even we set mu and sigma that is not very close to the “actual value”, they will converge to the it, as Kalman filter gathering more data.

3. Naïve Recalibration

b.

Result:

```
> optimal_naive
$`Withdrwal`
 [1]  16.55628  15.77460  18.89424  21.85478  25.84019  31.18718  26.61348  21.11574  24.14392  24.99108  26.08557
[12]  31.59387  32.57517  35.72315  39.99397  48.62324  50.13624  62.23529  49.19007  49.31723  52.81609  58.83476
[23]  66.37643  68.40674  80.63644  85.13711 112.35189 131.36205 133.66223 170.14691

$Remaining_value
 [1] 478.3362 542.1339 604.8063 696.4764 808.9708 678.8985 517.6196 581.6989 578.6926 570.2561 662.3141 652.2368
[13] 681.2052 731.3354 814.0755 778.5263 888.4096 661.4547 610.2988 586.7512 595.4839 605.9098 552.1359 551.9494
[25] 476.0823 496.1731 431.8890 285.3084 170.1469   0.0000
```

The first vector is the withdrawal amount in each year. The second vector is the remaining value of the founding in each year. **(Note: Initial asset, b0, is set as 500)**

Mean of withdrawal vector is

```
> mean(optimal_naive$Withdrwal)
[1] 53.7392
```

,

which greater than d0(16.55628).

c.

```
> optimal_naive
$`Withdrwal`
 [1] 16.55628 14.78187 16.59095 17.98287 19.92412 22.53359 18.01884 13.39684 14.35405 13.92268 13.61786 15.45547
[13] 14.93265 15.34514 16.09854 18.34032 17.72089 20.61301 15.26697 14.34317 14.39406 15.02526 15.88447 15.34012
[25] 16.94462 16.76449 20.73112 22.71345 21.65672 25.83325

$Remaining_value
 [1] 448.23325 476.04530 497.65562 537.01925 584.50355 459.65292 328.40272 345.83193 322.39305 297.69981 323.99883
[12] 298.98926 292.61669 294.38023 307.06304 275.17376 294.25093 205.29368 177.49610 159.90835 152.07509 144.99963
[23] 123.81570 115.98447  93.74616  91.55362  74.67674  46.22731  25.83325   0.00000
```

The first vector is the withdrawal amount in each year. The second vector is the remaining value of the founding in each year. **(Note: Initial asset, b0, is set as 500)**

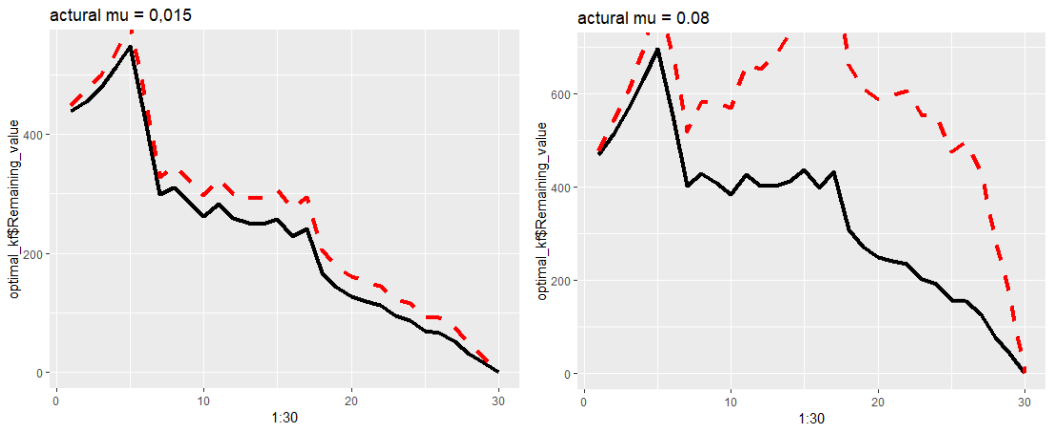
Mean of withdrawal vector is

```
> mean(optimal_naive$Withdrwal)
[1] 17.16946
```

,

which is greater than d0(16.55628).

Advantage of Kalman filter:



Black curve is the remaining value calculated with updated mu and sigma (Kalman), red dash line are remaining value calculated with fixed mu and sigma (Naive). Notice that the black line is much smother than the red one. It suggests the advantage of Kalman filter.