Challenging Independence of Company Valuations

Alvaro Callejas, Addis Gunst, Alexander Kaim, Jesse Pulselli

University of Massachusetts, Amherst

Department of Mathematics & Statistics

Stat 535, Fall 2019

Abstract

Given that the profits of a company are a function of revenues and costs, it is possible to think that the net present value (NPV) depends on two vectors of random variables where each entry represents the realization of revenue and cost at each period. This implies that the researcher will need to assign a probability distribution to the net present value (NPV), which suggests the construction of a statistical model that can adjust to the data generation process for the observable data. During this task, an information structure for the proposed model needs to be defined.

Our project aims to analyze how different information assumptions may lead to inconsistent parameter estimations. This is done by testing the per-

formance of a model that assumes temporal, or time-related, independence among revenue and cost realizations using the probability estimation of the simulated NPV realization. The simulated NPV realization comes from a data generation process that considers temporal correlation within revenue and cost. To accomplish this objective, this analysis begins by using a model that accounts for temporal correlation among revenue and cost used to simulate a specific NPV. Then, using a model that assumes identically independent distributed realizations of revenue and cost, we perform a Monte Carlo simulation and construct a likelihood distribution that allows us to recover the model parameters. With the parameters at hand, the joint probability of observing the simulated NPV is calculated and then compared with its actual probability calculated from the data generating process.

---

1 Introduction

In economics, it is widely accepted that the revenues and cost observed by one firm at a specific period depend on previous realizations of these variables. As a result, many consulting firms and investment companies use Autoregressive Models to predict firm performance based on past behavior [1]. In this context, using a model where each realization of revenue and cost is independent of previous realizations in company valuations may be a significant

departure from the nature of the phenomenon under study.

On the other hand, using a simple model that does not account for time correlation among realizations of random variables may bring simplicity and speed to the estimation. If this is the case, the reliability of this simple model will depend on the asymptotic nature of the estimators. If a simpler model leads to an unbiased estimator with a loss in efficiency, the researcher may have to choose between complexity and efficiency in the estimation [2]. But if the simpler model leads to a bias result, then the researcher will have to discard this model given that the results are not reliable.

In order to check whether time correlation among realization of random variables of a model may have important repercussions on the nature of the estimators, a simple net present value model is used. The NPV model depends on three variables: revenues B, costs C, and a discount factor δ:

$$NPV = B_0 - C_0 + \sum_{n=1}^{N} \frac{B_n - C_n}{(1+\delta_n)^n} \tag{1}$$

For all estimations, we assume that B and C are binary random variables [1] and independent from each other; the discount value is fixed to δ=0.04.

To begin our analysis, a set of B and C are simulated by using a model that accounts for temporal correlations among realizations. Using these variables, the simulated NPV of the firm is computed. Next, we define a model where B and C follow a Bernoulli distribution and all realizations are independent of each other. After, we perform a Monte Carlo simulation using the

[1] Revenue and cost can only take High and Low values.

defined model to construct a likelihood distribution for the simulated NPV and select the parameter that maximizes this distribution.

Using the maximum likelihood parameter, we compute the estimated probability of the simulated NPV under the simpler model and later compare it with the actual probability of the simulated NPV calculated under the simulation model.

To answer the question regarding the consistency of the probability estimator, we repeat this process a thousand times in order to create an empirical distribution for the relative difference between the estimated and the actual probability of the simulated NPV. By observing the shape of the empirical distribution, we can conclude if the estimations based on the simpler model may be consistent or not [3].

2 Methods

The methods used in this paper can be separated into three sections: Data simulation, Monte Carlo maximum likelihood estimation, and empirical density probability distribution estimation. The combination of these elements allow us to reach a conclusion on whether or not modeling a time correlated data using a time independent model may generate an inconsistent estimation of the probability of a firm's NPV.

The idea is to start with a simulation that accounts for temporal correlation between revenue and cost realizations. By construction of the NPV

formula, each simulation of costs and revenues is associated with a unique net present value. Thus, the probability of observing a particular NPV can be interpreted as the joint probability of observing a set of revenues and costs at each time period. This allows for the calculation of the probability of each NPV simulated.

The simulation process consists of a Markov chain process, in which the realization of each random variable at time t_{n+1} depends on its previous realization t_n. The model assumes independence between costs and revenues at each period and across time. The proposed model assumes that the transition probabilities for the Markov chain process are the same for revenue and cost, which means that this model depends on three parameters: probability of observing "high" (H) at time t_0 represented by P_H, and the two transition probabilities of going from "high" to "high" P_{HH}[2] and going from "low" (L) to "low" P_{LL}. Figure 1 represents the transition probabilities of moving from the realized state at t_n to the next state at t_{n+1}. Using this model compute the simulated NPV following the algorithm presented in Annex 1.

The probability of the simulated NPV results from the multiplication of the probability of the realization at t_0 (P_{s1}), and the transition probabilities in each time period (Conditional probability $P_{s2|s1}$) .

[2] P_{HH} is the conditional probability of seeing a High realization in time $(t+1)$ if we saw High in time t

$$P_{npv} = \left[P^B_{s1} * P^B_{s2|s1} * ... * P^B_{sn|sn-1}\right] * \left[P^C_{s1} * P^C_{s2|s1} * ... * P^C_{sn|sn-1}\right]^{3} \quad (2)$$

In a real application of a company valuation exercise, the researcher would be given a specific NPV and will be asked to estimate the probability of observing this outcome, given the potential future scenarios for the evaluated firm. In this case, one possible option to tackle this question is to choose a probabilistic model for the realization of revenue and cost at each period and perform a series of Monte Carlo simulations to create a likelihood distribution that allows for the identification of the model's parameters. We follow this approach to check the performance of a model that assumes time independence among revenue and cost.

The model used in the simulation assumes that the realizations of revenue and cost follow a binomial distribution with parameter p_H. This means that observing "high" revenue in time t_n brings no information regarding the state of this variable at time t_{n+1}.

With the model at hand, we perform the Monte Carlo simulation. Each simulation consists of randomly sampled set of revenue and cost realizations from a binomial distribution with parameter p_H and then we use these realizations to compute the NPV using equation (1). We vary the value of p_H in increments of 0.1 from 0 to 1 and perform $10,000$ simulations for each

[3]Subscript *st* represents the state at time t, which can be high or low.

value. Then, we record how many times we observe the simulated NPV on the Monte Carlo simulation of each value of p_H, which allows for the construction of the likelihood distribution. Finally, we select the model with the highest likelihood and use that model to calculate the estimated probability of the simulated NPV. Now we can compare the actual probability of the simulated NPV P_{npv} and the estimated probability of the simulated NPV $\hat{P}_{npv}$.

$$\hat{P}_{npv} = \left[P_{s1}^{B} * P_{s2}^{B} * ... * P_{sn}^{B}\right] * \left[P_{s1}^{C} * P_{s2}^{C} * ... * P_{sn}^{C}\right] \tag{3}$$

We repeat this process of simulating an NPV and estimating the probability of the simulated NPV using Monte Carlo simulations 1,000 times. This allows for the construct of a probability density function of the percent difference between P_{npv} and $\hat{P}_{npv}$ by using Kernel Density estimation. The estimator used in the construction of the empirical probability function is the following:

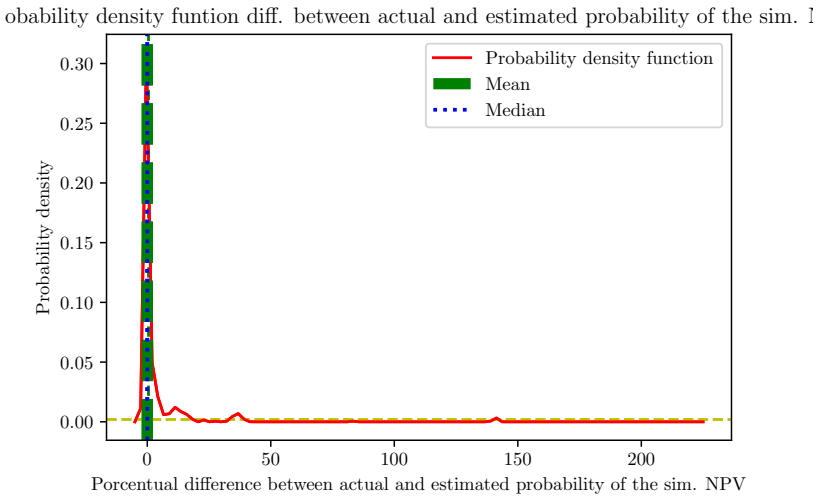
$$diff = \frac{\hat{P}_{npv} - P_{npv}}{P_{npv}} \tag{4}$$

The intuition behind using this estimator for the construction of the probability density function is that if the time-independent model produces a consistent estimator of the probability of the simulated NPV the density probability function should be centered at 0. A deviation from these results may imply that the used model is not generating a consistent estimator.

3 Results

From these results, one can conclude that the results from assuming time independence in the likelihood model may lead to an unbiased but somehow inefficient estimator of the probability of observing a specific NPV. An example of a simulation that resulted in the overestimation of the probability of the simulated NPV can be seen in Figure 2.

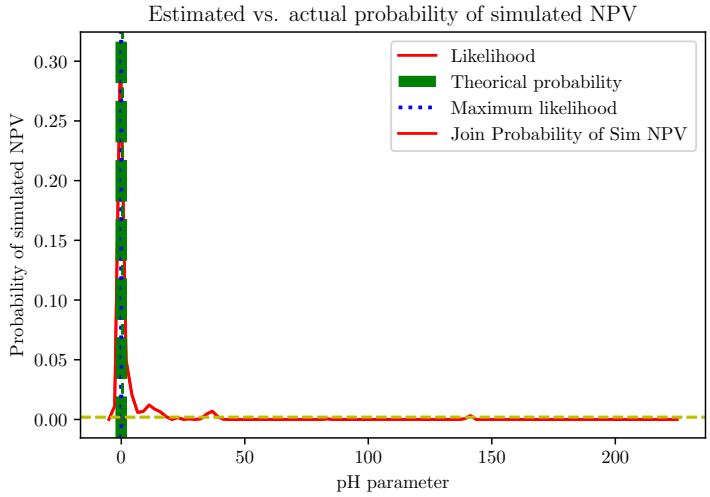
The results below in Figure 1 represent our Monte Carlo simulation:



From Figure 1 above, we can conclude that assuming independence in the likelihood estimation model is useful as long as there is substantial data to estimate the model. This figure of our estimated probability function has the maximum likelihood centered around 0.5, which suggests that the estimator of the time independent model may produce a consistent estimator. With a higher number of simulations, (say 1,000,000), we'd eventually see the peaks

of likelihood and theoretical probability line up as well. In our Monte Carlo simulation we conducted 10,000 simulations for each possible value of p_h to find our independent model.

In Figure 2 below, it can be observed that the probability density function allocates probability to positive differences between $\hat{P}_{npv}$ and P_{npv} as a result of our chosen parameters.



The density, mean, and median are centered at 0; therefore, there is generally no percentual difference between the *actual* probability of the simulated NPV and the *estimated* probability of the simulated NPV. This situation suggests that the time independent model produces an estimator that is unbiased.

When acting as a company choosing between the models to conduct valuation, the more likely one to be chosen is the independence model because of

it's computational simplicity resulting in a quicker running time. For these reasons, using the independence model could save a company time and money in the long run.

Our results suggest that assuming independence in the likelihood estimation model may be useful if there is a substantial amount of data to estimate the model. When data is scarce, it is a better idea to work with a more sophisticated model that assumes time correlation among random variables.

4 Summary & Discussion

In conclusion of the works done in this paper, the claim that was being tested was that assuming intertemporal independence of costs and revenues could potentially bias the results of a probabilistic company valuation. Through a series of various simulations and results that were produced and examined as explained in some of the above sections, there was some closure to the question at hand. We can confidently say that a time-independent model produces and unbiased estimator, but may also not be very reliable due to the large variance observed. There may be some remedial measures taken to increase the efficiency though, such as re-running the Monte Carlo simulations many times to recover a parameter with reduced variance. But in the scenario of this project, there will be some opportunity cost to be considered when deciding between models to conduct the company valuation.

References

[1] Jason Brownlee. Autoregression Models for Time Series Forecasting With Python, https://machinelearningmastery.com/autoregression-models-time-series-forecasting-python/ 2019

[2] Rajat Harlalka. Choosing the Right Machine Learning Algorithm, https://hackernoon.com/choosing-the-right-machine-learning-algorithm-68126944ce1f 2018

[3] Eran Raviv. Bias vs. Consistency, https://eranraviv.com/bias-vs-consistency/ 2014

Appendix

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Nov 18 16:51:51 2019

@author: user
"""

"""
STAT535 NPV is Life
Alvaro Callejas
Addis Gunst
Alexander Kaim
Jesse Pulselli
"""

import random
import numpy as np
import matplotlib.pyplot as plt
import time
```

```
def sim(p=0.5, ph=0.7, pl=0.8, n=20,sd=70):
    """Funtion that simulates data, ph represents probability of going f
    high to high, pl represents probability of going from low to low, n
    length of the realization, p is probability of observing high on t0
    random.seed(sd)
    te=np.array(random.sample(range(1000),n-1))/1000#random draw
    random.seed(sd)
    rea=random.choices('HL',weights=(p,1-p),k=1) #initial realization a
    if rea[0]=='H':
        jp=p
    else:
        jp=round((1-p),2)
    for i in range(n-1): #compute realization from t1 to tn
        temp=te[i]
        if rea[-1]=='L': #using probability of low to low to be compared
            if temp<=pl:
                rea.append('L')
                jp=jp*pl
            else:
                rea.append('H')
                jp=jp*round((1-pl), 2)
        else: #using probability of high to high to be compared with dra
            if temp<=ph:
```

```
            rea.append('H')
            jp=jp*ph
        else:
            rea.append('L')
            jp=jp*round((1-ph), 2)
    return (rea, jp)


def ind_sim(n,CV,BV,N,p,d):
    """Function that simulates data where the variables are independent
    dic={}
    dic2={}
    for i in range(N):
        Bt=random.choices('HL', weights=(p,1-p), k=n)
        pb=[round((1-p), 5) if x=='L' else p for x in Bt]
        Ct=random.choices('HL', weights=(p,1-p), k=n)
        pc=[round((1-p), 5) if x=='L' else p for x in Ct]
        [npvt,pr]=NPV(Bt,Ct,BV,CV,d,np.prod(pb),np.prod(pc))
        if npvt in dic.keys():
            dic[npvt] += 1
        else:
            dic[npvt] = 1
            dic2[npvt] =pr
    return (dic, dic2)
```

```
def dep_sim(real,npr,BV,CV,n):
    """Function that calculate the theoretical probability asuming depe
    among diferent realizations"""
    s_npv=""
    pb1=1
    pc1=1
    while s_npv!=real:
        Bt=random.choices('HL', weights=(.5,.5), k=n)
        Ct=random.choices('HL', weights=(.5,.5), k=n)
        [s_npv,ppr]=NPV(Bt,Ct,BV,CV,d,pb1,pc1)
    pb=join_p(Bt,npr)
    pc=join_p(Ct,npr)
    return (pb*pc)

random.seed(5)
seed_r=np.array(random.sample(range(10000),1000))
seed_c=np.array(random.sample(range(10000),1000))
prob=[i/100 for i in range(0,101,10)]
n=6
it=0
d=0.04 #discount rate
BV=[12000, 36000] #revenue values
```

```
CV=[6000, 18000] #cost values


lkp=[]
rea_pro=[]

start_time=time.time()
for j in range(len(seed_r)):
    print(j)
    sdd_r=seed_r[j]
    sdd_c=seed_c[j]
    [B,pb]=sim(p=0.5, ph=0.7, pl=0.8, n=n,sd=sdd_r) #simulation of rever
    [C,pc]=sim(p=0.5, ph=0.7, pl=0.8, n=n,sd=sdd_c) #simulation of cost
    [real, pnpv]=NPV(B,C,BV,CV,d,pc,pb)
    f=[]
    jo_p=[]
    sim_num=10000
    for pr in prob:
        [x,y] = ind_sim(n,CV,BV, sim_num, pr,d)
        if real in x:
            f.append(x[real]/sim_num)
            jo_p.append(y[real])
        else:
```

```
                f.append(0)
                jo_p.append(None)
        lkp.append(jo_p[np.argmax(f)])
        rea_pro.append(pnpv)

print(time.time()-start_time)
```